

Cache me if you can: Capacitated Selfish Replication in Networks*

Ragavendran Gopalakrishnan
California Institute of Technology
ragad3@caltech.edu

Dimitrios Kanoulas
Northeastern University
dkanou@ccs.neu.edu

Naga Naresh Karuturi
Google Inc.
naganareshk@google.com

C. Pandu Rangan
IIT Madras, India
rangan@iitm.ernet.in

Rajmohan Rajaraman
Northeastern University
rraj@ccs.neu.edu

Ravi Sundaram
Northeastern University
koods@ccs.neu.edu

December 13, 2011

Abstract

Motivated by peer-to-peer (P2P) networks and content delivery applications, we study Capacitated Selfish Replication (CSR) games, which involve nodes on a network making strategic choices regarding the content to replicate in their caches. Selfish replication games were introduced in [6], who analyzed the uncapacitated case leaving the capacitated version as an open direction.

In this work, we study pure Nash equilibria of CSR games with an emphasis on hierarchical networks, which have been extensively used to model the communication costs of content delivery and P2P systems. The best result from previous work on CSR games for hierarchical networks [21, 27] is the existence of a Nash equilibrium for a (slight generalization of a) 1-level hierarchy when the utility function is based on the sum of the costs of accessing the replicated objects in the network. Our main result is an exact polynomial-time algorithm for finding a Nash Equilibrium in any hierarchical network using a new technique which we term “fictional players”. We show that this technique extends to a general framework of natural preference orders, orders that are entirely arbitrary except for two natural constraints - “Nearer is better” and “Independence of irrelevant alternatives”. This axiomatic treatment captures a vast class of utility functions and even allows for nodes to simultaneously have utility functions of completely different functional forms.

Using our axiomatic framework, we next study CSR games on arbitrary networks and delineate the boundary between intractability and effective computability in terms of the network structure, object preferences, and the total number of objects. In addition to hierarchical networks, we show the existence of equilibria for general undirected networks when either object preferences are binary or there are two objects. For general CSR games, however, we show that it is NP-hard to determine whether equilibria exist. We also show that the existence of equilibria in strongly connected networks with two objects and binary object preferences can be determined in polynomial time via a reduction to the well-studied even-cycle problem. Finally, we introduce a fractional version of CSR games (F-CSR) with application to content distribution using erasure codes. We show that while every F-CSR game instance possesses an equilibrium, finding an equilibrium in an F-CSR game is PPAD-complete.

*Gopalakrishnan and Karuturi were partially supported by a generous gift from Northeastern University alumnus Madhav Anand. This work was also partially supported by NSF grants CCF-0635119 and CNS-0915985.

1 Introduction

Consider a P2P movie sharing service where you need to decide which movies to store locally, given your limited disk space, and which to obtain from your friends. Note that your decisions affect those of your friends, who in turn take actions that affect you. A natural question arises: what is the prognosis for you and your network of friends in terms of the stability of your movie collections and the satisfaction you will derive from them? Similarly, in the brave new wireless world of 4G you will not only be a consumer of different apps, you (your personal communications and computing device) will also be a provider of apps to others around you. And the question arises: could this lead to a situation of endless churn (in terms of what apps to store) or could there be an equilibrium?

In this paper, we study Capacitated Selfish Replication (CSR) Games, which provide an abstraction of the above scenarios. These are games in which the strategic agents, or players, are nodes in a network. The nodes have object preferences as well as bounded storage space – caches – in which they can store copies of the content. Each node cooperates with other nodes by serving their requests to access objects stored in its cache. However, the set of objects that a node chooses to store in its cache is entirely based on its own utility function and where objects of interest have been stored in the network.

Such a game-theoretic framework was first introduced by Chun et al [6], who analyzed pure Nash equilibria in a setting with storage costs but no cache capacities, and left the capacitated version as an open direction. Recent work on CSR games has focused on *hierarchical networks*, which have been extensively used to model the communication costs of content delivery and P2P systems. (For instance, see the seminal work of [14] that uses the ultrametric model for content delivery networks and the work of [22, 16, 17, 32] on cooperative caching in hierarchical networks.) The best result from previous work on CSR games for hierarchical networks [21, 27] is the existence of a Nash equilibrium for (a slight generalization of) a one-level hierarchical network using the sum utility function, i.e., when the utility of each node is based on a weighted sum of the cost of accessing the objects.

1.1 Our results

This paper studies the existence and computability of Nash equilibria for several variants of CSR games, with a particular focus on hierarchical networks. As with earlier studies [6, 21, 27, 1], we focus on the case where all pieces of content have the same size; note that otherwise even computing the best response of a player (node) is a generalization of the well-known knapsack problem and is NP-hard.

- Our main result is a polynomial-time algorithm for finding a Nash equilibrium for CSR games in any hierarchical network, thus resolving the question left open by [19, 20, 27]. Our algorithm, presented in Section 3, is based on a new technique that we call the method of “fictional players¹” where we introduce and eliminate fictional players iteratively in a controlled fashion, maintaining a Nash equilibrium at each step, until the end when we have the desired equilibrium for the entire network (without any fictional players).

The above result is presented specifically in the context of the sum utility function to elucidate the technique of fictional players. We then abstract the central requirements for our proof technique and develop a general axiomatic framework which allows us to extend our results to a large class of utility functions.

- We present, in Section 4, a general framework for CSR games involving utility preference relations and node preference orders. Rather than specifying a numerical utility assigned by each node to each placement of objects, we only require that the preference order each node has on object placements

¹not to be confused with “fictitious play” [10] which involves learning

Object preferences and count	Undirected networks	Directed networks
Binary, two objects	Yes, in P (5.3)	No, in P (6.2)
Binary, three or more objects	Yes, in PLS (5.2)	No, NP-complete (6.1)
General, two objects	Yes, in P (5.3)	No, NP-complete (6.1)
General, three or more objects	No, NP-complete (6.1) Hierarchical: Yes, in P (5.1)	No, NP-complete (6.1)

Table 1: Existence and computability of equilibria in CSR games. Each cell (other than in the first row or the first column) first indicates whether equilibria always exist in the particular sub-class of CSR games. If equilibria always exist, then the cell next indicates the complexity of determining an equilibrium; otherwise, it indicates the complexity of determining whether equilibria exist for a given instance. The relevant subsection is given in parentheses.

satisfy two natural constraints of Monotonicity (or “Nearer is better”) and Consistency (or “Independence of irrelevant alternatives”). This axiomatic treatment captures a vast class of utility functions and even allows for nodes to simultaneously have utility functions of completely different functional forms.

- We extend our result for hierarchical networks to the broader class of utilities allowed by the axiomatic framework, and then study general CSR games obtained by considering different network structures (directed or undirected) and different forms of object preferences (binary or general). We delineate the boundary between intractability and effective computability of equilibria in terms of the network structure, object preferences, and the total number of objects. These results are presented in Sections 5 and 6 and summarized in Table 1. Notable results include: (1) the existence of equilibria for undirected networks with two objects that also utilizes the technique of fictional players, (2) the existence of equilibria for undirected networks when object preferences are binary, and (3) the equivalence of finding equilibria in CSR games with two objects and binary object preferences to the well-studied even-cycle problem [29].

Our last set of results concerns fractional CSR (F-CSR) games, where each node is allowed to store fractions of objects. In our framework, a node can satisfy an object access request by retrieving any set of fractions of the object as long as these fractions sum to at least one. As we discuss in Section 7, a natural implementation of this framework is via erasure codes (e.g., using the Digital Fountain approach [4, 31]).

- We show that F-CSR games always have equilibria, and the problem of finding an equilibrium is in PPAD. We also show, however, that finding equilibria is PPAD-hard even for a sum-of-distances utility function.

1.2 Related work

In the last decade there has been a tremendous flowering of research at the intersection of game theory and computer science [24]. In a seminal paper [26] Papadimitriou laid the groundwork for algorithmic game theory by introducing syntactically defined subclasses of FNP with complete problems, PPAD being a notable such subclass. Recently, in a major breakthrough 2-player Nash Equilibrium was shown to be PPAD-complete [7, 5]. The term PPAD-complete is coming to occupy a role in algorithmic game theory analogous to the term NP-complete in combinatorial optimization [11].

Selfish caching games were introduced in [6] who considered the uncapacitated case where nodes could store more pieces of content by paying for the additional storage. We believe that limits on cache-capacity model an important real-world restriction and hence our focus on the capacitated version which was left as

an open direction by [6]. Special cases of the integral version of CSR games have been studied. In [21], Nash equilibria were shown to exist for when nodes are equidistant from one another and a special server holds all objects. [27] slightly extends [21] to the case where special servers for different objects are at different distances. Our results generalize and completely subsume all these prior cases of CSR games. The Market sharing games defined by [12] also consider caches with capacity, but are of a very special kind; unlike CSR games, market sharing games are a special case of congestion games. In this work we focus primarily on equilibria and our general axiomatic framework has the flavor of similar frameworks from the theory of social choice [2]; in this sense, we deviate from prior work [9, 8] that is focused on the price of anarchy [18].

There has been considerable research on capacitated caching, viewed as an optimization problem. Various centralized and distributed algorithms have been presented for different networks in [1, 3, 22, 16, 33].

2 A basic model for CSR games

We consider a network consisting of a set V of nodes labeled 1 through $n = |V|$ sharing a collection O of unit-size objects. For any i and j in V , let d_{ij} denote the cost incurred at i for accessing an object at j ; we refer to d as the access cost function. We say that j is node i 's *nearest* node in a set S of nodes if j is in S and $d_{ij} \leq d_{ik}$ for all k in S . We say that the given network is *undirected* if d is symmetric; that is, if $d_{ij} = d_{ji}$ for all i, j in V . We call an undirected network *hierarchical* if the access cost function forms an ultrametric; that is, if $d_{ik} \leq \max\{d_{ij}, d_{jk}\}$ for all $i, j, k \in V$.

Each node i has a cache to store a certain number of objects. The placement at a node i is simply the set of objects stored at i . The strategy set of a given node is the set of all feasible placements at the node. A *global placement* is any tuple $(P_i : i \in V)$, where $P_i \subseteq O$ represents a feasible placement at node i . For convenience, we use P_{-i} to denote the collection $(P_j : j \in V \setminus \{i\})$, thus often using $P = (P_i, P_{-i})$ to refer to a global placement. We also assume that V includes a (server) node that has the capacity to store all objects. This ensures that at least one copy of every object is present in the system; this assumption can be made without loss of generality since we can set the access cost of every node to this server to be arbitrarily large.

CSR Games. In our game-theoretic model, each node attaches a utility to each global placement. We assume that each node i has a weight $r_i(\alpha)$ for each object α representing the rate at which i accesses α . We define the *sum utility function* $U_s(i)$ as follows: $U_s(i)(P) = -\sum_{\alpha \in O} r_i(\alpha) \cdot d_{i\sigma_i(P, \alpha)}$, where $\sigma_i(P, \alpha)$ is i 's nearest node holding α in P .

A CSR game is a tuple $(V, O, d, \{r_i\})$. Our focus is on *pure Nash equilibria* (henceforth, simply *equilibria*) of the CSR games we define. An equilibrium for a CSR game instance is a global placement P such that for each $i \in V$ there is no placement Q_i such that $U_s(i)(P) > U_s(i)(Q_i)$.

Unit cache capacity. In this paper, we assume that all objects are of identical size. Under this assumption, we now argue that it is sufficient to consider the case where each node's cache holds exactly one object. Consider a set V of nodes in which the cache of node i can store c_i objects. Let V' denote a new set of nodes which contains, for each node i in V , new nodes i_1, i_2, \dots, i_{c_i} , i.e., one new node for each unit of the cache capacity of i . We extend the access cost function as follows: $d_{j_\ell i_k} = d_{ji}$ for all $1 \leq \ell \leq c_j$, $1 \leq k \leq c_i$.

We consider an obvious onto mapping f from placements in V' to those in V . Given placement P' for V' , we set $f(P') = P$ where $P_i = \cup_{1 \leq k \leq c_i} P'_{i_k}$. This mapping ensures that $U_s(i)(P') = U_s(i)(P)$, giving us the desired reduction. Thus, in the remainder of the paper, we assume that every node in the network stores at most one object in its cache.

3 Hierarchical networks

In this section, we give a polynomial-time construction of equilibria for CSR games on hierarchical networks. Any hierarchical network can be represented by a tree T whose set of leaves is the node set V and every internal node v has a label $\ell(v)$ such that (a) if v is an ancestor² of w in T , then $\ell(v) \geq \ell(w)$, and (b) for any i, j in V , d_{ij} is given by $\ell(\text{lca}(i, j))$, where $\text{lca}(i, j)$ denotes the least common ancestor of nodes i and j [14, 16].

Fictional players. In order to present our algorithm, we introduce the notion of a *fictional player*. For an object α , a *fictional α -player* is a new node that stores α in any equilibrium; for any fictional α -player ℓ , $r_\ell(\alpha)$ is 1 and $r_\ell(\beta)$ is 0 for any $\beta \neq \alpha$. Each fictional player is introduced as a leaf in the current hierarchy; the exact locations in the hierarchy are determined by our algorithm. The access cost function is naturally extended to the fictional players using the hierarchy and the labels of the internal nodes. In the following, we use “node” to refer to both the elements of V and fictional players.

A preference relation. The hierarchical network and the weights that nodes have for different objects induce, for each node i , a natural preorder \sqsubseteq_i among elements of $\mathbf{O} \times A_i$, where A_i is the set of proper ancestors of i in T . Specifically, we define $(\alpha, v) \sqsubseteq_i (\beta, w)$ whenever $r_i(\alpha) \cdot \ell(v) > r_i(\beta) \cdot \ell(w)$. We can now express the best response of any player directly in terms of these preference relations. We define $\mu_i(P) = (\alpha, v)$ where $P_i = \{\alpha\}$ and v is $\text{lca}(i, \sigma_i(P_{-i}, \alpha))$, where $\sigma_i(P_{-i}, \alpha)$ denotes i ’s nearest node in the set of nodes holding α in P_{-i} .

Lemma 1. *A best response P_i of a node i for a placement P_{-i} of $V \setminus \{i\}$ is $\{\alpha\}$ where α maximizes $(\gamma, \text{lca}(i, \sigma_i(P_{-i}, \gamma)))$, over all objects γ , according to \sqsubseteq_i .*

Proof. For a given placement P with $P_i = \{\alpha\}$, $U_s(i)(P)$ equals $-\sum_{\gamma \neq \alpha} r_i(\gamma) \ell(\text{lca}(i, \sigma_i(P_{-i}, \gamma)))$, which can be rewritten as $-(\sum_{\gamma \in \mathbf{O}} r_i(\gamma) \ell(\text{lca}(i, \sigma_i(P_{-i}, \gamma))) + r_i(\alpha) \cdot \ell(\text{lca}(i, \sigma_i(P_{-i}, \alpha)))$. Thus, $\{\alpha\}$ is a best response to P_{-i} if and only if α maximizes $r_i(\gamma) \cdot \ell(\text{lca}(i, \sigma_i(P_{-i}, \gamma)))$ over all objects γ . The desired claim follows from the definition of \sqsubseteq_i . \square

The algorithm. We introduce several fictional players at the start of the algorithm. We maintain the invariant that the current global placement is an equilibrium in the current hierarchy. As the algorithm proceeds, the set of fictional players and their locations change as we remove existing fictional players or add new ones. On termination, there are no fictional players leaving us with a desired equilibrium. Let W_t and P^t denote the set of fictional players and equilibrium, respectively, at the start of step t of the algorithm.

Initialization. We add, for each object α and for each internal node v of T , a fictional α -player as a leaf child of v ; this constitutes the set W_0 . The initial equilibrium P^0 is defined as follows: for each fictional α -player i , we have $P_i^0 = \{\alpha\}$; each node i in V plays its best response. Clearly, each fictional player is in equilibrium, by definition. Furthermore, for every α , every i in V has a sibling fictional α -player. Thus, the best response of every i in V is independent of the placement of nodes in $V \setminus \{i\}$, implying that P^0 is an equilibrium.

Step t of algorithm. Fix an equilibrium P^t for the node set $V \cup W_t$. If W_t is empty, then we are done. Otherwise, select a node j in W_t . Let $P_j^t = \{\alpha\}$, and let $\mu_j(P^t) = (\alpha, v)$. Let S denote the set of all nodes $i \in V$ such that $(\alpha, v) \sqsubseteq_i \mu_i(P^t)$. We now describe how to compute a new set of fictional players W_{t+1} and a new global placement P^{t+1} such that P^{t+1} is an equilibrium for $V \cup W_{t+1}$. We consider two cases.

- S is empty: Remove the fictional player j from W_t and the hierarchy, and leave the placement in the remaining nodes as before. Thus $W_{t+1} = W_t - \{j\}$ and P^{t+1} is the same as P^t except that P_j^{t+1} is no longer defined.

²We adopt the convention that each node is both descendant and ancestor of itself.

- S is nonempty: Select a node i in S such that $\text{lca}(i, j)$ is lowest among all nodes in S . Let $P_i^t = \{\beta\}$. We set $P_i^{t+1} = \{\alpha\}$, remove the fictional α -player j from W_t , and add a new fictional β -player ℓ as a leaf sibling of i in T ; i.e., $P_\ell^{t+1} = \{\beta\}$. For every other node j , set $P_j^{t+1} = P_j^t$. Finally, set $W_{t+1} = (W_t \cup \{k\}) \setminus \{j\}$.

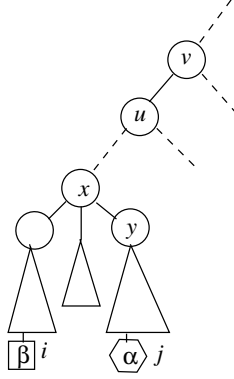


Figure 1: Illustrating the algorithm for hierarchical networks; the square is a node i in V holding object β , and the hexagon is a fictional α -player.

Lemma 2. Consider step t of the algorithm. If P^t is an equilibrium for $V \cup W_t$, then the following statements hold.

1. For every node k in $V \cup W_{t+1}$, P_k^{t+1} is a best response to P_{-k}^{t+1} .
2. For every node k in $V \cup W_{t+1}$, $\mu_k(P^{t+1}) \supseteq_k \mu_k(P^t)$.
3. We have $|W_{t+1}| \leq |W_t|$. Furthermore, either $|W_{t+1}| < |W_t|$ or there exists a node i in V such that $\mu_i(P^{t+1}) \supseteq_i \mu_i(P^t)$.

Proof. Let α, v, S, i , and j be as defined in step t of the algorithm above, and illustrated in Figure 1. We first establish statements 1 and 2 of the lemma. Let k be any node in $V \cup W_{t+1}$. Consider first the case where $\text{lca}(k, j)$ is an ancestor of v (i.e., k is not in the subtree rooted at the child u of v that contains j). For any object γ , we have $\sigma_k(P_{-k}^{t+1}, \gamma) = \sigma_k(P_{-k}^t, \gamma)$ and $P_k^{t+1} = P_k^t$. It thus follows that $\mu_k(P^{t+1}) = \mu_k(P^t)$, implying statement 2 for k . Since P^t is in equilibrium, statement 1 also holds for k .

We next establish statements 1 and 2 for any node k where $\text{lca}(k, j)$ is a proper descendant of v (i.e., k is in the subtree rooted at the child u of v that contains j). We consider two cases. The first case is where S is empty. In this case, the fictional α -player j is removed; thus j is not in W_{t+1} . Furthermore, there is no copy of α in the subtree rooted at u . Since no object other than α is created or removed in this case, we have $\sigma_k(P_{-k}^{t+1}, \gamma) = \sigma_k(P_{-k}^t, \gamma)$ for $\gamma \neq \alpha$. We also have $\text{lca}(k, \sigma_k(P_{-k}^{t+1}, \alpha)) = v$ and $\mu_k(P^{t+1}) = \mu_k(P^t)$, the latter establishing statement 2 for k . Since S is empty, $\mu_k(P^t) \supseteq_k (\alpha, v)$. It follows from Lemma 1 and the fact that P_k^t is in equilibrium that P_k^{t+1} is a best response against P_{-k}^{t+1} , establishing statement 1 for k .

The second case is where S is not empty. Let i be as defined above, i.e., i is a node in S such that $\text{lca}(i, j)$ is lowest among all nodes in S . Let x denote $\text{lca}(i, j)$. Let P_i^t be equal to $\{\beta\}$, where $\beta \neq \alpha$. By the algorithm, we have $P_k^{t+1} = \{\alpha\}$. Let $k \neq i$ be a node in the subtree rooted at u . For any $\gamma \neq \alpha$, $\sigma_k(P_{-k}^{t+1}, \gamma) = \sigma_k(P_{-k}^t, \gamma)$. Since $P_k^{t+1} = P_k^t \neq \{\alpha\}$, we have $\mu_k(P^{t+1}) = \mu_k(P^t)$, establishing statement 2 for k . For node i , we have $\mu_i(P^{t+1}) = (\alpha, v) \supseteq_i \mu_i(P^t)$, establishing statement 2 for i .

It remains to establish statement 1 for any node k in the subtree rooted at u . We again separate into two cases. Let y be the child of x that is an ancestor of j (see Figure 1(c)). In the first case, we let k be in the

subtree rooted at y . Then, by our choice of i , we have

$$\mu_k(P^{t+1}) \sqsupseteq_k (\alpha, v) \sqsupseteq_k (\alpha, x) = (\alpha, \sigma_k(P_{-k}^{t+1}, \alpha)),$$

which, by Lemma 1, implies that statement 1 holds for k . In the second case, we let k be in the subtree rooted at u but not in the subtree rooted at y . Again, $\sigma_k(P_{-k}^{t+1}, \gamma) = \sigma_k(P_{-k}^t, \gamma)$ for $\gamma \neq \alpha$. And for α we have

$$(\alpha, \text{lca}(k, \sigma_k(P_{-k}^{t+1}, \alpha))) = (\alpha, \text{lca}(k, i)) \sqsupseteq_k (\alpha, x) \sqsupseteq_k \mu_k(P^t) = \mu_k(P^{t+1}),$$

establishing statement 1 for k using Lemma 1.

We finally establish statement 3 of the lemma. The fact $|W_{t+1}| \leq |W_t|$ is immediate from the definition of step t of the algorithm. When S is empty, $|W_{t+1}| < |W_t|$ since a fictional player is deleted. When S is nonempty, we have shown above that $\mu_i(P^{t+1}) \sqsubset_i \mu_i(P^t)$, thus completing the proof for statement 3 and for the whole lemma. \square

Theorem 3. *For hierarchical node preferences, an equilibrium can be found in polynomial time.*

Proof. It is immediate from the definition of the algorithm and Lemma 2 that at termination, the algorithm returns a valid equilibrium. It remains to show that our algorithm terminates in polynomial time. Consider the potential given by the sum of $|W_t|$ and the sum, over all i , of the position of $\mu_i(P^t)$ in the preorder \sqsupseteq_i . The term $|W_0|$ is at most nm , where n is $|V|$ (which is at least the number of internal nodes) and m is the number of objects. Furthermore, since $|O \times I|$ is at most nm , the initial potential is at most $nm + n^2m$. By Lemma 2, the potential decreases by at least one in each step of the algorithm. Thus, the number of steps of the algorithm is at most $nm + n^2m$.

We now show that each step of the algorithm can be implemented in polynomial time. The initialization consists of adding the $O(nm)$ fictional players and computing the best response for each node i in V ; the latter task involves, for each k in V , comparing at most m placements (one for each object). Each subsequent step of the algorithm involves the selection of a fictional player j , determination whether the set S is nonempty, and if so, computation of the node i , and then updating the placement. The only parts that need explanation are the computation of S and $i - S$ is simply the set of all nodes k that are not in equilibrium when fictional player j is deleted. We compute S as follows: for each node k in V , if replacing the current object in their cache by α yields a more preferable placement (according to the utility) then add k to S . Thus, S can be computed in time polynomial in n . The node i is simply a node in S such that $\text{lca}(i, j)$ is lowest among all nodes in S , and can be computed in time polynomial in n . This completes the proof of the theorem. \square

4 A general axiomatic framework for CSR games

We now present a new axiomatic framework which generalizes the result of Section 3 to a broad class of utility functions, and also enables us to study the existence and complexity of equilibria in more general settings.

Node preference relations. We assume that each node i in V has a total preorder \geq_i among all the nodes in V^3 ; \geq_i further satisfies $i \geq_i j$ for all $i, j \in V$. We say that a node i *prefers* j over k if $j \geq_i k$, and call a node j *most i -preferred* in a set S of nodes if j is in S and $j \geq_i k$ for all k in S . We also use the notation $j =_i k$ whenever $j \geq_i k$ and $k \geq_i j$, and $j >_i k$ whenever it is not the case that $k \geq_i j$. Note that $>_i$ is

³A total preorder is a binary relation that satisfies reflexivity, transitivity, and totality. Totality means that for any i, j, k , either $j \geq_i k$ or $k \geq_i j$.

a strict weak order⁴, and for any i, j , and k , we have exactly one of these three relations holding: $j >_i k$, $k >_i j$, $k =_i j$. We also extend the notation $\sigma_i(P, \alpha)$ and $\sigma_i(P_{-i}, \alpha)$ denote a most i -preferred node holding α in P and P_{-i} , respectively, breaking ties arbitrarily.

The access cost function d introduced in Section 2 induces a natural node preference relation: $j >_i k$ if $d_{ij} < d_{ik}$, and $j =_i k$ if $d_{ij} = d_{ik}$. In fact, as we show in Lemma 4, undirected networks (i.e., when the access cost function is symmetric) are equivalent to acyclic node preference collections. Formally, the collection $\{\geq_i: i \in V\}$ is an *acyclic node preference collection* if there does not exist a sequence of nodes i_0, i_1, \dots, i_{k-1} for an integer $k \geq 3$ such that $i_{(j-1) \bmod k} >_{i_j} i_{(j+1) \bmod k}$ for all $0 \leq j < k$.

Lemma 4. *Any undirected network yields an acyclic node preference collection. For any acyclic node preference collection, we can compute, in polynomial time, symmetric cost functions that are consistent with the node preferences.*

Proof. Let d denote a symmetric access cost function over the set V of nodes. For a given node $i \in V$, we have $j \geq_i k$ iff $d_{ij} \leq d_{ik}$. We now argue that the collection $\{\geq_i: i \in V\}$ is acyclic. Suppose, for the sake of contradiction, that there exists a sequence of nodes i_0, i_1, \dots, i_{k-1} for an integer $k \geq 3$ such that $i_{(j-1) \bmod k} >_{i_j} i_{(j+1) \bmod k}$ for all $0 \leq j < k$. It then follows that:

$$d_{i_j i_{(j-1) \bmod k}} < d_{i_j i_{(j+1) \bmod k}} \text{ for } 0 \leq j < k.$$

Since d is symmetric, we obtain

$$d_{i_j i_{(j-1) \bmod k}} < d_{i_{(j+1) \bmod k} i_j} \text{ for } 0 \leq j < k,$$

which is a contradiction.

Given an acyclic collection of node preferences, we compute an associated access cost function d in polynomial time as follows. We construct a directed graph G over the set U of all unordered pairs $(i, j) : i, j \in V, i \neq j$. There is a directed edge from node (i, j) to (i, k) if and only if $k \geq_i j$. Since the collection $\{\geq_i: i \in V\}$ is acyclic, G is a dag. We compute the topological ordering $\pi : U \rightarrow \mathbb{Z}$; thus, we have $\pi((i, j)) < \pi((k, \ell))$ whenever there is a directed path from (i, j) to (k, ℓ) . Setting d_{ij} to be $\pi((i, j))$ gives us the desired undirected network. \square

Utility preference relations. In our game-theoretic model, each node attaches a utility to each global placement. We present a general definition that allows us to consider a large class of utility functions simultaneously. Rather than define a numerical utility function, we present the utility at each node i as a total preorder \succeq_i among the set of all global placements. (The notation \succ_i and $=_i$ over global placements are defined analogously.) We require that \succeq_i , for each $i \in V$, satisfies the following two basic conditions.

- **Monotonicity:** For any two global placements P and Q , if, for each object α and each node q with $\alpha \in Q_q$, there exists a node p with $\alpha \in P_p$ and $p \geq_i q$, then $P \succeq_i Q$.
- **Consistency:** Let (P_i, P_{-i}) and (Q_i, Q_{-i}) denote two global placements such that for each object $\alpha \in P_i \cup Q_i$, if p (resp., q) is a most i -preferred node in $V \setminus \{i\}$ holding α , i.e., $\alpha \in P_p$ (resp., $\alpha \in Q_q$), then $p =_i q$. If $(P_i, P_{-i}) \succ_i (Q_i, P_{-i})$, then $(P_i, Q_{-i}) \succeq_i (Q_i, Q_{-i})$.

In words, the monotonicity condition says that for any node, if all the objects in a placement are placed at nodes that are at least as preferred as in another placement, then the node prefers the former placement at least as much as the latter. The consistency condition says that the preference for a node to store one set of

⁴A strict weak order is a strict partial order $>$ (a transitive relation that is irreflexive) in which the relation “neither $a > b$ nor $b > a$ ” is transitive. Strict weak orders and total preorders are widely used in microeconomics.

objects instead of another is entirely a function of the set of most preferred other nodes that together hold these objects. For instance, if a node i with unit capacity prefers to store α over β in a scenario where the most i -preferred node (other than i) storing α (resp., β) is j (resp., k), then i prefers to store α at least as much as β in any other situation where the most i -preferred node (other than i) storing α (resp., β) is j (resp., k).

Generality of the conditions. We note that many standard utility functions defined for replica placement problems [6, 20, 27], including the sum and max functions, satisfy the monotonicity and consistency conditions. Indeed, any utility function that is an L_p norm, for any p , over the costs for the individual objects, also satisfies the conditions. Furthermore, since the monotonicity and consistency conditions apply to the individual utility functions, our model allows the different nodes to adopt different types of utilities, as long as each separately satisfies the two conditions.

Binary object preferences. One class of utility preference relations that we highlight is the ones based on binary object preferences. Suppose that each node i has a set S_i of objects in which it is equally interested, and it has no interest in the other objects. Let $\tau_i(P)$ denote the $|S_i|$ -length sequence consisting of the $\sigma_i(P, \alpha)$, for $\alpha \in S_i$, in nonincreasing order according to the relation \succeq_i . Then, the consistency condition can be further strengthened to the following.

- **Binary Consistency:** For any placements $P = (P_i, P_{-i})$ and $Q = (Q_i, Q_{-i})$ with $P_{-i} = Q_{-i}$, we have $P \succeq_i Q$ if and only if for $1 \leq k \leq |S_i|$, the k th component of $\tau_i(P)$ is at least as i -preferred as the k th component of $\tau_i(Q)$.

CSR Games. In the general framework, a CSR game is a tuple $(V, O, \{\succeq_i\}, \{\succeq_i\})$. A (pure) Nash equilibrium for an CSR game instance is a global placement P such that for each $i \in V$ there is no placement Q_i such that $(Q_i, P_{-i}) \succ_i (P_i, P_{-i})$.

For our complexity results, we need to give the specification for a given game instance. The set V is specified, together with node cache capacities, and O is an enumerated list of object names. The node preference relation \succeq_i is specified succinctly by a set of at most $\binom{n}{2}$ bits, for each i . The utility preference relation \succeq_i , however, is over a potentially exponential number of placements (in terms of n , m , and cache sizes). For our complexity results, we assume that the utility preference relations are specified by an efficient algorithm – which we call the *utility preference oracle* – that takes as input a node i , and two global placements P and Q , and returns whether $P \succeq_i Q$. For the sum, max, and L_p -norm utilities, the utility preference oracle simply computes the relevant utility function. For binary object preferences, the binary consistency condition yields an oracle that is polynomial in number of nodes, objects, and cache sizes.

Unit cache capacity. We now argue that the unit cache capacity assumption of Section 2 continues to hold without loss of generality. Consider a set V of nodes in which the cache of node i can store c_i objects. Let V' denote a new set of nodes which contains, for each node i in V , new nodes i_1, i_2, \dots, i_{c_i} , i.e., one new node for each unit of the cache capacity of i . We set the node preferences as follows: for all $i, i', j \in V$, $1 \leq f, \ell \leq c_j$, $1 \leq k, k' \leq c_i$, we have $i_k \succeq_{j_\ell} i'_{k'}$ whenever $i \succeq_j i'$, and $j_\ell =_{i_k} j_\ell$.

We consider an obvious onto mapping f from placements in V' to those in V . Given placement P' for V' , we set $f(P') = P$ where $P_i = \cup_{1 \leq k \leq c_i} P'_{i_k}$. This mapping naturally defines the utility preference relations for the node set V' . In particular, for any $i \in V$ and $1 \leq k \leq c_i$, $P' \succeq_{i_k} Q'$ whenever $f(P') \succeq_i f(Q')$. We also note that f is computable in time polynomial in the number of nodes and the sum of the cache capacities. It is easy to verify that the utility preference relation \succeq_{i_k} for all $i_k \in V'$ satisfies the monotonicity and consistency conditions. Furthermore, P' is an equilibrium for V' if and only if $f(P')$ is an equilibrium for V ; this together with the onto property of the mapping f gives us the desired reduction.

5 Existence of equilibria in the general framework

In this section, we establish the existence of equilibria for several CSR games under the axiomatic framework of Section 4. We first extend our result for the sum utility function on hierarchical networks to the general framework (Section 5.1). We next show that CSR games for undirected networks and binary object preferences are potential games (Section 5.2). Finally, when there are only two objects in the system, we use the technique of fictional players to give a polynomial-time construction of equilibria for CSR games for undirected networks (Section 5.3).

5.1 Hierarchical networks

We now show that the polynomial time algorithm of Section 3 extends to the axiomatic framework we have introduced. In the general framework, a hierarchical network can be represented as a tree T whose set of leaves is the node set V and the node preference relation \geq_i given by: $j \geq_i k$ if $\text{lca}(i, j)$ is a descendant of $\text{lca}(i, k)$. Our algorithm of Section 3 and its analysis are completely determined by the structure of the hierarchical network and the pair-preference relations \sqsupseteq_i defined for each node i ; the latter were defined for the sum utility function. In order to extend our analysis to the general framework, it suffices to derive a new preference relation and establish the analogue of Lemma 1, which we now present for arbitrary utility preference relations satisfying the monotonicity and consistency properties.

Pair preference relations. Given any utility preference relation \succeq_i that satisfies the monotonicity and consistency conditions, we define a strict weak order \sqsubset_i on $O \times A_i$, where A_i is the set of proper ancestors of i in T .

1. For each object α , node i , and proper ancestors v and w of i , we have $(\alpha, v) \sqsubset_i (\alpha, w)$ whenever v is a proper ancestor of w .
2. Consider distinct objects α, β and nodes i, j, k with $j, k \neq i$. Let \mathcal{P} denote the set of global placements P such that j (resp., k) is a most i -preferred node in $V \setminus \{i\}$ holding α (resp., β) in P_{-i} . If there exist global placements $P = (\{\alpha\}, P_{-i})$ and $Q = (\{\beta\}, P_{-i})$ in \mathcal{P} with $P \succ_i Q$, then $(\alpha, \text{lca}(i, j)) \sqsubset_i (\beta, \text{lca}(i, k))$.

In words, item 1 says that i 's preference for keeping α in its cache increases as the most i -preferred node holding α becomes less preferred (or “moves farther away”). In item 2, $(\alpha, v) \sqsubset_i (\beta, w)$ means that if i needs to place either α or β in its cache, and the least common ancestor of i and the most i -preferred node in $V \setminus \{i\}$ holding α (resp., β) is v (resp., w), then i prefers to store α over β . The strict weak order \sqsubset_i induces a total preorder \sqsupseteq_i as follows: $(\alpha, v) \sqsupseteq_i (\beta, w)$ if it is not the case that $(\beta, v) \sqsubset_i (\alpha, w)$. We similarly define $=_i$: $(\alpha, v) =_i (\beta, w)$ if $(\alpha, v) \sqsupseteq_i (\beta, w)$ and $(\beta, v) =_i (\alpha, w)$.

Lemma 5. For each i , \sqsubset_i as given above, is a well-defined strict weak order.

Proof. We need to ensure the well-definedness of part 2 of the definition of pair preference relations. That is, we need to show that for any placements P_{-i} and Q_{-i} such that a most i -preferred node in P_{-i} holding α (resp., β) is also a most i -preferred node in Q_{-i} , it is impossible that $(\{\alpha\}, P_{-i}) \succ_i (\{\beta\}, P_{-i})$ and $(\{\beta\}, Q_{-i}) \succ_i (\{\alpha\}, Q_{-i})$ both hold. This directly follows from the consistency condition for utility preference relations.

The reflexivity and transitivity of \sqsupseteq_i are immediate from the definitions and the reflexivity and transitivity of \succeq_i . Finally, to ensure the well-definedness of the strict preorder \sqsubset_i , we also have to show that there is no collection of pairs (α_j, v_j) , $0 \leq j < \ell$ for some integer $\ell > 1$, such that $(\alpha_j, v_j) \sqsubset_i (\alpha_{j+1 \bmod \ell}, v_{j+1 \bmod \ell})$ for $0 \leq j < \ell$. To see this, it is sufficient to note that if $(\alpha, v) \sqsubset_i (\alpha', v')$ then for all placements P and P' such that $P_{-i} = P'_{-i}$ and the least common ancestor of i and the most i -preferred

node in $V \setminus \{i\}$ that holds α (resp., α') is v (resp., v') we have $P \succ_i P'$. So any cycle in the strict preorder \sqsubset_i implies a cycle in \succ_i , yielding a contradiction. \square

Lemma 6. *For any global placement $P = (\{\alpha\}, P_{-i})$, if j (resp., k) is a most i -preferred node holding object α (resp., β) in P_{-i} , and $(\{\beta\}, P_{-i}) \succ_i (\{\alpha\}, P_{-i})$, i.e., for node i , storing β is a better response to P_{-i} than storing α , then $(\beta, \text{lca}(i, k)) \sqsubset_i (\alpha, \text{lca}(i, j))$. Furthermore, for any global placement P , $\{\alpha\}$ is a best response to P_{-i} , where α maximizes $(\gamma, \text{lca}(i, \sigma_i(P_{-i}, \gamma)))$, over all objects γ , according to \sqsubset_i .*

Proof. The first statement of the lemma directly follows from item 2 of the definition of pair preference relations. We establish the second statement by contradiction. Suppose that for node i , $\{\beta\}$ is a better response to P_{-i} than $\{\alpha\}$. Then, we have $(\{\beta\}, P_{-i}) \succ_i (\{\alpha\}, P_{-i})$, which, by item 2 of the definition of pair preference relations, implies that $(\beta, \text{lca}(i, \sigma_i(P_{-i}, \beta))) \sqsubset_i (\alpha, \text{lca}(i, \sigma_i(P_{-i}, \alpha)))$, a contradiction to the choice of α . \square

The remainder of the analysis for hierarchical networks (Lemma 2 and Theorem 3) follows as before, invoking Lemma 6 instead of Lemma 1.

5.2 Undirected networks with binary object preferences

Let d be a symmetric cost function for an undirected network over the node set V . Recall that for binary object preferences, we are given, for each node i a set S_i of objects in which i is equally interested. Our proof of existence of equilibria is via a potential function argument. Given a placement P , let $\Phi_i(P) = d_{ij}$, where j is the most i -preferred node in $V \setminus \{i\}$ holding the object in P_i . We introduce the potential function Φ : $\Phi(P) = (\Phi_0, \Phi_{i_1}(P), \Phi_{i_2}(P), \dots, \Phi_{i_n}(P))$, where Φ_0 is the number of nodes i such that $P_i \subseteq S_i$, and $\Phi_{i_j}(P) \leq \Phi_{i_{j+1}}(P)$, $\forall j$, where $V = \{i_1, i_2, \dots, i_n\}$. We prove that Φ is an increasing potential function: after any better response step, Φ increases in lexicographical order.

Let $P = (P_i, P_{-i})$ be an arbitrary global placement. Assume that $P_i = \{\alpha\}$ and j is the most i -preferred node in P_{-i} holding α . Consider any better response step, from placement P to $Q = (Q_i, P_{-i})$, where $Q_i = \{\beta\}$. Clearly $\beta \in S_i$. We consider two cases. First, suppose $\alpha \notin S_i$ and $\beta \in S_i$. Then, Φ_0 increases, and so does the potential. The second case is where $\alpha, \beta \in S_i$. Let k be the most i -preferred node in P_{-i} holding β . In this case, Φ_0 does not change. However, since this is a better response step of i , $j \succ_i k$, implying that $d_{ik} > d_{ij}$ and hence $\Phi_i(Q) > \Phi_i(P)$. Consider any other node j . If j holds any object γ other than β , since no new copy of γ has been added, $\Phi_j(Q) \geq \Phi_j(P)$. It remains to consider the case where j holds β . If S is the set of nodes in $V \setminus \{j\}$ holding β in P_{-j} , then $S \cup \{i\}$ is the set of nodes in $V \setminus \{j\}$ holding β . Thus, $\Phi_j(Q) = \min\{\Phi_j(P), d_{ji}\} \geq \min\{\Phi_j(P), \Phi_i(Q)\}$. This also means that $\Phi_j(P)$ appears later in the sorted order than $\Phi_i(P)$ and $\Phi_j(Q)$ appears no earlier in the sorted order than $\Phi_i(Q)$. Hence, $\Phi(Q)$ is lexicographically greater than $\Phi(P)$. This establishes that for undirected networks with binary object preferences, the resulting CSR game is a potential game, and hence also in PLS [13].

5.3 Undirected networks with two objects

We give a polynomial-time algorithm for computing an equilibrium in any undirected network with two objects. Our algorithm uses the fictional player technique introduced in Section 5.1. It starts by introducing fictional players serving both the objects in the network at zero cost from each node. In each subsequent step, we move the fictional players progressively “further” away, ensuring that each instant, we have an equilibrium. Finally, when the fictional players are at least preferred cost from all the nodes, they can be removed yielding an equilibrium for the original network.

Suppose we are given a undirected network with access cost function d . Also let \mathcal{D} be the set $\{0, \ell_1, \ell_2, \dots, \ell_r\}$ of all access costs between nodes in the system in increasing order; that is, $\ell_1 = \min_{i,j} d_{ij}$ and $\ell_r = \max_{i,j} d_{ij}$ and $\ell_i < \ell_{i+1}$ for all $1 \leq i < r$.

Fictional player. For an object α , a fictional α -player is a new node that will store α in every equilibrium; an fictional α -player prefers storing α over any other object. We denote by $srv_\alpha(\ell)$ the fictional α -player which is at access cost ℓ from every node in V .

The algorithm.

Initialization. Assuming that there are two objects α and β in the system, we initially set up a fictional α -player $srv_\alpha(0)$ and β -player $srv_\beta(0)$ at access cost 0 from each node in V . We let nodes replicate their most preferred object and access the other without any access cost from the corresponding fictional player. This placement is obviously an equilibrium.

Step t of algorithm. Fix an equilibrium P for the node set $V \cup \{srv_\alpha(\ell_t)\} \cup \{srv_\beta(\ell_t)\}$. We describe one step of the algorithm which computes a new set of fictional players $srv_\alpha(\ell_{t+1})$ and $srv_\beta(\ell_{t+1})$ and a new placement P' such that P' is an equilibrium for the node set $V \cup \{srv_\alpha(\ell_{t+1})\} \cup \{srv_\beta(\ell_{t+1})\}$. We first remove the α -player $srv_\alpha(\ell_t)$ from the system and instead we add $srv_\alpha(\ell_{t+1})$. If there do not exist nodes that want to deviate we are done. Otherwise, assume that there exists a node i that wants to deviate from its strategy. Since the most i -preferred node holding β in $V \cup \{srv_\alpha(\ell_t)\} \cup \{srv_\beta(\ell_t)\}$ remains the same in $V \cup \{srv_\alpha(\ell_{t+1})\} \cup \{srv_\beta(\ell_t)\}$, i is not holding object α . Thus the only nodes that may want to deviate are those that are holding object β . We argue that if we let i to deviate from $\beta \in P_i$ to $\alpha \in P'_i$, there is no node $j \in V \setminus \{i\}$ that gets affected by i 's deviation. Consider the following two cases:

- If a node j has access cost at most ℓ_t from i , then $\beta \in P_j$. Otherwise, if $\alpha \in P_j$, $srv_\alpha(\ell_t)$ would not be the most i -preferred node holding α and thus i would not be affected by any change of α -players. Thus there does not exist any node $j \in V \setminus \{i\}$ with access cost at most ℓ_t from i , such that $\alpha \in P_j$, and as we showed above $\alpha \in P'_j$.
- If a node j has access cost at least ℓ_{t+1} from i , then $P_j = P'_j$. Because of the α -player $srv_\alpha(\ell_{t+1})$ and the β -player $srv_\beta(\ell_t)$, i would never be the j -most preferred node in P' .

We then remove the β -player $srv_\beta(\ell_t)$ from the system and instead we add $srv_\beta(\ell_{t+1})$. Using a similar argument as above, we obtain a new equilibrium at the end of this step.

Theorem 7. *For undirected networks with two objects, an equilibrium can be found in polynomial time.*

Proof. An initial placement P , where we have the set of fictional players $srv_\alpha(0)$ and $srv_\beta(0)$ in the system, is obviously an equilibrium. It is immediate from our argument above that at termination the algorithm returns a valid equilibrium.

The size of the set \mathcal{D} is at most $\binom{n}{2}$ which is at most n^2 . In each step t at most n nodes may want to deviate from their strategy, since we showed above that if a node deviates once in a step, it will not deviate again during the same step. Thus, the total number of deviations in the algorithm is at most n^3 . \square

6 Non-Existence of equilibria in CSR games and the associated decision problem

In this section, we show that the classes of games studied in Section 5 are essentially the only games where equilibria are guaranteed to exist. We identify the most basic CSR games where equilibria may not exist, and study the complexity of the associated decision problem.

6.1 NP-Completeness

The main result of this section is the following theorem, which establishes the NP-hardness of determining whether a given CSR game has an equilibrium even when the utility preference relations are based on the

sum utility function and either the number of objects is small or the object preferences are binary. The proof is by a polynomial-time reduction from 3SAT [11]. Each reduction is built on top of a gadget which has an equilibrium if and only if a specified node holds a certain object. Several copies of these gadgets are then put together to capture the given 3SAT formula.

Theorem 8. *The problem of determining whether a given CSR instance has an equilibrium is in NP. It is NP-hard to determine whether an CSR instance has an equilibrium even if one of these three restrictions hold: (a) number of objects is two; (b) object preferences are binary and number of objects is three; (c) network is undirected and number of objects is three.*

The membership in NP is immediate, since one can determine in polynomial time whether a given global placement is an equilibrium. The remainder of the proof focuses on the hardness reduction from 3SAT.

Given a 3SAT formula ϕ with n variables x_1, x_2, \dots, x_n and k clauses c_1, c_2, \dots, c_k , we construct an CSR instance as follows. For each variable x_i in ϕ , we introduce two variable nodes: node X_i and \bar{X}_i . We set $d_{X_i \bar{X}_i}$ and the symmetric $d_{\bar{X}_i X_i}$ to be 0.5, where d is the underlying access cost function. For each clause c_j , we introduce a clause node C_j . Assuming that $\ell_{j,r}$, for $r \in \{1, 2, 3\}$ are the three literals of clause c_j in formula ϕ , we set $d_{C_j \ell_{j,r}}$ and $d_{\ell_{j,r} C_j}$ to be 1 for $r \in \{1, 2, 3\}$. Note that each $\ell_{j,s}$, for $j \in [1, k]$, and $s \in \{1, 2, 3\}$ is in fact some variable node X_h or \bar{X}_h for some $h \in [1, n]$. We also introduce a gadget G illustrated in Figure 3, consisting of nodes S, A, B , and C . We set the access cost d_{SC_i} and the symmetric $d_{C_i S}$, for all $1 \leq i \leq k$ between node S and all clause nodes to be 2. The general construction is illustrated in Figure 2.

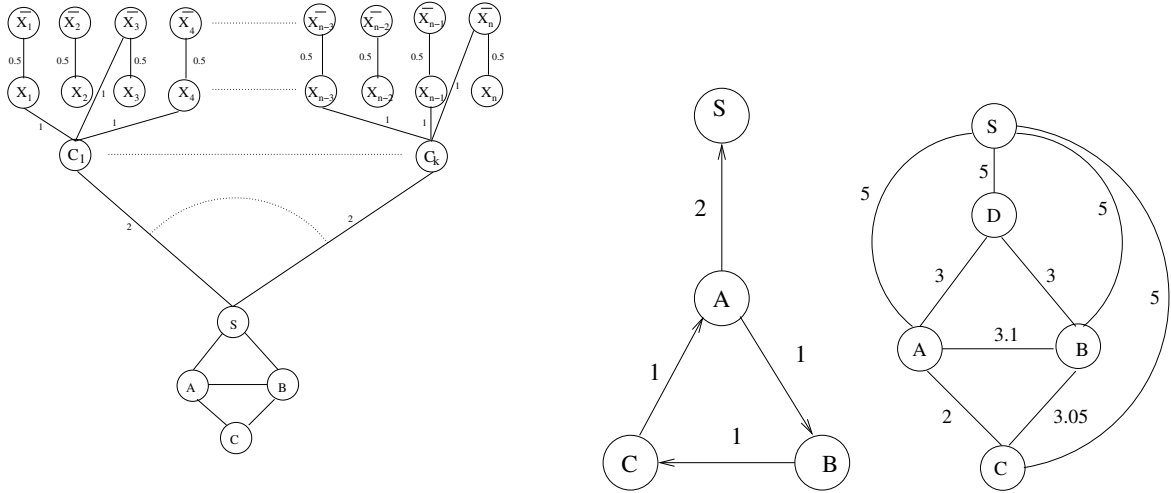


Figure 2: Instance of the construction for the undirected

case proof of NP-Hardness, where $\phi = (x_1 \vee \bar{x}_3 \vee x_4) \wedge \dots \wedge (x_{n-3} \vee x_{n-1} \vee \bar{x}_n)$

Figure 3: Gadget for the directed (left) and the undirected (right) case

Directed networks with two objects. We set $d_{AS}, d_{AB}, d_{BC}, d_{CA}$ to be 1. The server node at access cost $d_{srv} = 10$ from all nodes in V , stores a fixed copy of two objects α and β . We set the weights $r_{x_i}(\alpha), r_{\bar{x}_i}(\alpha), r_{x_i}(\beta), r_{\bar{x}_i}(\beta)$ of each variable node on objects α and β , to be 1. For each clause node we set the weight $r_{C_i}(\alpha)$ on object α to be 1, and the weight $r_{C_i}(\beta)$ on object β to be 0.7, for all $1 \leq i \leq k$. We set the weight $r_S(\alpha)$ on object α to be 1, and the weight $r_S(\beta)$ on object β to be 0.7. We set the weight $r_A(\alpha)$ of node A on object α to be 0.7 and the weight $r_A(\beta)$ of node A on object β to be 1. Finally, we set the weight of nodes B and C on both objects α , and β to be 1. We refer to this CSR instance as I_1 .

Undirected networks with three objects. For the undirected case, we set d_{AS} and d_{BS} to be 3, d_{AB} to be 3.1, d_{BC} to be 3.05, and d_{CA} to be 2; while symmetry holds. The server node, which is at access cost

$d_{srv} = 5$ from all nodes in V , stores a fixed copy of three objects α , β , and γ . We set the weights $r_{x_i}(\alpha)$, $r_{\bar{x}_i}(\alpha)$, $r_{x_i}(\beta)$, $r_{\bar{x}_i}(\beta)$ of each variable node on objects α and β , to be 1. Also, for each clause node we set the weight $r_{C_i}(\alpha)$ on object α to be 0.85, and the weight $r_{C_i}(\beta)$ on object β to be 1, for all $1 \leq i \leq k$. We set the weight $r_S(\alpha)$ on object α to be 0.85, and the weight $r_S(\beta)$ on object β to be 1. Also we set the weight $r_A(\alpha) = 1$, $r_A(\gamma) = 2$, $r_B(\beta) = 1$, $r_B(\gamma) = 0.9837$, $r_C(\beta) = 1$, and $r_C(\gamma) = 1.6$. We set all the remaining weights to be 0. We refer to this CSR instance as I_2 .

Lemma 9. *A variable node X_i holds object α (resp., β) if and only if node \bar{X}_i holds object β (resp., α).*

Proof. The proof is immediate, since \bar{X}_i (resp., X_i) is X_i 's (resp., \bar{X}_i 's) nearest node, and both X_i and \bar{X}_i are interested equally in α and β . \square

Lemma 10. *Clause node C_i holds object α if and only if its variable nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold object β .*

Proof. First, assume that $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold β . These nodes are C_i 's nearest nodes holding β . By Lemma 9 we know that nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ hold α , and they are C_i 's nearest nodes holding α . Node's C_i cost for holding α and accessing β from $L_{i,j}$, for $j \in \{1, 2, 3\}$, is $r_{C_i}(\beta)d_{C_i L_{i,j}} = 1 \cdot 1 = 1$; while the cost for holding β and accessing α from $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$, is $r_S(\alpha)d_{C_i \bar{L}_{i,j}} = 0.85 \cdot 1.5 = 1.275$. Obviously, node C_i prefers to replicate α .

Now assume that at least one of the nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ holds α . These nodes are C_i 's nearest nodes holding α . Also, by Lemma 9, C_i 's nearest nodes holding β are all the remaining nodes from the set $L_{i,j}$, $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$, that don't hold α . Node's C_i cost for holding β and accessing α from $L_{i,j}$, for $j \in \{1, 2, 3\}$, is $r_{C_i}(\alpha)d_{C_i L_{i,j}} = 0.85 \cdot 1 = 0.85$; while the cost for holding α and accessing β from node $\bar{L}_{i,j}$ (resp., $L_{i,j}$), is $r_{C_i}(\beta)d_{C_i \bar{L}_{i,j}} = 1 \cdot 1.5 = 1.5$ (resp., $r_{C_i}(\beta)d_{C_i L_{i,j}} = 1 \cdot 1 = 1$). Obviously, in any case node C_i prefers to replicate β . \square

Lemma 11. *Node S holds object α if and only if all clause nodes C_1, \dots, C_k hold object β .*

Proof. First, assume that C_1, \dots, C_k are holding β . These nodes are S 's nearest nodes holding β . Also by Lemma 10, S 's nearest node holding α is at least one of $L_{i,j}$ nodes, where $i \in [1, k]$, $j \in \{1, 2, 3\}$. The cost for S holding α and accessing β from a node C_i , $i \in [1, k]$, is $r_S(\beta)d_{S C_i} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1, k]$, $j \in \{1, 2, 3\}$, is $r_S(\alpha)d_{S L_{i,j}} = 0.85 \cdot 3 = 2.55$. Obviously, node S prefers to replicate α .

Now assume that at least one of C_1, \dots, C_k holds α . These nodes are S 's nearest node holding α . Also S 's nearest node holding β , due to Lemma 10 is one of $L_{i,j}$, where $i \in [1, k]$, $j \in \{1, 2, 3\}$. The cost for holding β and accessing α from a node C_i , is $r_S(\alpha)d_{S C_i} = 0.85 \cdot 2 = 1.7$; while the cost for holding α and accessing β from a node $L_{i,j}$, where $j \in \{1, 2, 3\}$, is $r_S(\beta)d_{S L_{i,j}} = 1 \cdot 3 = 3$. Obviously, in any case node S prefers to replicate β . \square

Theorem 12. *The CSR instance I_1 has an equilibrium if and only if node S holds object α .*

Proof. First, assume that S is holding α . By Lemma 11 nodes C_1, \dots, C_k hold object β , and by Lemma 10 at least one of nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ for each node C_i , $i \in [1, k]$, holds object α , and the corresponding $\bar{L}_{i,j}$ is holding object β . We claim that the placement where A holds β , B holds β , and C holds α , is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy.

Node A does not want to deviate since its cost for holding object β and accessing α from A 's nearest node S , is $r_A(\alpha)d_{AS} = 0.7 \cdot 1 = 0.7$; while the cost for holding object α and accessing β from A 's nearest node B , is $r_A(\alpha)d_{AB} = 1 \cdot 1 = 1$. Node B does not want to deviate since its cost for holding object β and accessing α from B 's nearest node C , is $r_B(\alpha)d_{BC} = 1 \cdot 1 = 1$; while the cost for holding object α and accessing β from B 's nearest node A , is $r_B(\beta)d_{BA} = 1 \cdot 2 = 2$. Node C does not want to deviate since

its cost for holding object α and accessing β from C 's nearest node A , is $r_C(\beta)d_{CA} = 1 \cdot 1 = 1$; while the cost for holding object β and accessing α from C 's nearest node S , is $r_C(\alpha)d_{CS} = 1 \cdot 2 = 2$. Also note that none of $S, C_1, \dots, C_k, L_{ij}, \bar{L}_{ij}$ for $i \in [1, k], j \in \{1, 2, 3\}$ is getting affected of the objects been holded by the gadget nodes.

Now assume that node S holds object β . We are going to prove that for every possible placement over nodes A, B , and C , at least one node wants to deviate from its strategy. Consider the following cases:

- Nodes A, B , and C hold object α : Node B (resp., C) wants to deviate, since the cost for holding object α and accessing β from B 's (resp., C 's) nearest node S , is $r_B(\beta)d_{BS} = 1 \cdot 3 = 3$ (resp., $r_C(\beta)d_{CS} = 1 \cdot 2 = 2$); while the cost for holding object β and accessing α from B 's nearest node A , is $r_B(\beta)d_{BA} = 1 \cdot 2 = 2$ (resp., $r_C(\beta)d_{CA} = 1 \cdot 1 = 1$).
- Two nodes hold object α and the third holds β : In the case where A and B hold α , A wants to deviate since the cost while holding α and accessing β from A 's nearest node S is $r_A(\beta)d_{AS} = 1 \cdot 1 = 1$; while the cost for holding β and accessing α from A 's nearest node B is $r_A(\alpha)d_{AB} = 0.7 \cdot 1 = 0.7$. In the case where A and C hold α , then C wants to deviate since the cost while holding α and accessing β from C 's nearest node B is $r_C(\beta)d_{CB} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from C 's nearest node A is $r_C(\alpha)d_{CA} = 1 \cdot 1 = 1$. In the case where B and C hold α , B wants to deviate since the cost while holding α and accessing β from B 's nearest node A is $r_B(\beta)d_{BA} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from B 's nearest node C is $r_B(\alpha)d_{BC} = 1 \cdot 1 = 1$.
- One node holds α : If A (resp., B , or C) holds α , B (resp., C , A) wants to deviate since the cost while holding β and accessing α from B 's (resp., C 's, or A 's) nearest node A (resp., B , or C) is $r_B(\alpha)d_{BA} = 1 \cdot 2 = 2$ (resp., $r_C(\alpha)d_{CB} = 1 \cdot 2 = 2$, or $r_A(\alpha)d_{AC} = 0.7 \cdot 2 = 1.4$); while the cost for holding α and accessing β from B 's (resp., C 's, or A 's) nearest node C (resp., A , or B), is $r_B(\beta)d_{BC} = 1 \cdot 1 = 1$ (resp., $r_C(\beta)d_{CA} = 1 \cdot 1 = 1$, or $r_A(\beta)d_{AB} = 1 \cdot 1 = 1$).
- Nodes A, B , and C hold β : All of them want to deviate. Node A wants to deviate since the cost while holding β and accessing α from A 's nearest node C_i , for some $i \in [1, k]$, is $r_A(\beta)d_{AC_i} = 1 \cdot 3 = 3$; while the cost for holding β and accessing α from A 's nearest node S is $r_A(\alpha)d_{AS} = 0.7 \cdot 1 = 0.7$. Similar proof holds for nodes B and C .

Obviously the system does not have a pure Nash equilibrium, which completes the proof. \square

Theorem 13. *The CSR instance I_2 has an equilibrium if and only if node S holds object α .*

Proof. First, assume that S is holding α . By Lemma 11 nodes C_1, \dots, C_k hold object β , and by Lemma 10 at least one of nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ for each node $C_i, i \in [1, k]$, holds object α , and the corresponding $\bar{L}_{i,j}$ is holding object β . We claim that the placement where A holds γ , node B holds β , and C holds γ is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy. Node A doesn't want to deviate since the cost for holding object γ and accessing object α from node S is $r_A(\alpha)d_{AS} = 3$; while the cost for holding α and accessing γ from node C increases to $r_A(\gamma)d_{AC} = 4$. Node B doesn't want to deviate since the cost for holding object β and accessing object γ from node C is $r_B(\gamma)d_{BC} = 0.9837 \cdot 3.05 = 3.000285$; while the cost for holding object β and accessing γ from the server increases to $r_B(\gamma)d_{srv} = 5$. Node C doesn't want to deviate since the cost for holding object γ and accessing β from node B is $r_C(\beta)d_{CB} = 3.05$; while the cost for holding object β and accessing γ from node A increases to $r_C(\beta)d_{CA} = 3.2$.

Now assume that node S holds object β . We are going to prove that for every possible placement over nodes A, B , and C , at least one node wants to deviate from its strategy. Consider the following cases:

- Node A holds α , node B holds γ , and node C holds β : Node A wants to deviate since the cost while it is holding object α and accessing object γ from node B is $(r_A(\gamma)d_{AB} = 6.2)$; while the cost for holding object γ and accessing α from the server decreases to $r_A(\alpha)d_{srv} = 5$.
- Node A holds γ , node B holds γ , and node C holds β : Node B wants to deviate since the cost while it is holding object γ and accessing object β from node C is $(r_B(\beta)d_{BC} = 3.05)$; while the cost for holding object β and accessing γ from node A decreases to $r_B(\gamma)d_{BA} = 3.04947$.
- Node A holds γ , node B holds β , and node C holds β : Node C wants to deviate since the cost while it is holding object β and accessing object γ from node A is $(r_C(\gamma)d_{CA} = 3.2)$; while the cost for holding object γ and accessing β from node B decreases to $r_C(\beta)d_{CB} = 3.05$.
- Node A holds γ , node B holds β , and node C holds γ : Node A wants to deviate since the cost while it is holding object γ and accessing object α from the server is $(r_A(\alpha)d_{srv} = 5)$; while the cost for holding object α and accessing γ from node C decreases to $r_A(\gamma)d_{AC} = 4$.
- Node A holds α , node B holds β , and node C holds γ : Node B wants to deviate since the cost while it is holding object β and accessing object γ from node C is $(r_B(\gamma)d_{BC} = 3.000285)$; while the cost for holding object γ and accessing β from node S decreases to $r_B(\beta)d_{BS} = 3$.
- Node A holds α , node B holds γ , and node C holds γ : Node C wants to deviate since the cost while it is holding object γ and accessing object β from the server is $(r_C(\beta)d_{srv} = 5)$; while the cost for holding object β and accessing γ from B decreases to $r_C(\gamma)d_{BC} = 4.88$.
- Node A holds α , node B holds β , and node C holds β : Node C wants to deviate since the cost while it is holding object β and accessing object γ from the server is $(r_C(\gamma)d_{srv} = 4.9185)$; while the cost for holding object γ and accessing β from node B decreases to $r_B(\beta)d_{CB} = 3.05$.
- Node A holds γ , node B holds γ , and node C holds γ : Node A wants to deviate since the cost while it is holding object γ and accessing object α from the server is $(r_A(\alpha)d_{srv} = 5)$; while the cost for holding object α and accessing γ from C decreases to $r_A(\gamma)d_{AC} = 4$.

The remaining placements where A holds α , B holds α , and C holds α , obviously are not stable since none of the nodes are interested in these objects. Since there does not exist a stable placement, an equilibrium does not exist. \square

Binary object preferences over three objects.. For the binary object preferences, we introduce two extra nodes K and L . We set $d_{C_i K}$, for $i \in [1, k]$, between clause nodes and K to be 1.4, d_{SL} to be 2.1, and d_{AS} , d_{AB} , d_{BC} , d_{CA} to be 1. The server node, which is at access cost $d_{srv} = 10$ from all nodes in V , stores a fixed copy of three objects α , β , and γ . Each node i has a set S_i of objects in which it is equally interested. For nodes X_i , \bar{X}_i , for $i \in [1, n]$, we set $S_{X_i} = \{\alpha, \beta\}$ and $S_{\bar{X}_i} = \{\alpha, \beta\}$. For nodes C_i , for $i \in [1, k]$, we set $S_{C_i} = \{\alpha, \gamma\}$. For node K we set $S_K = \{\gamma\}$; while for node L we set $S_L = \{\beta\}$. For node S we set $S_S = \{\alpha, \beta\}$. For nodes A , B , and C we set S_A , S_B , and S_C correspondingly to be the set $\{\alpha, \gamma\}$. As we mentioned in the binary object preference definition for our utility function $U_s(i)$, equally interested means weight 1 for all objects in S_i , and 0 for the remaining. We refer to this instance as I_3 .

Lemma 9 holds as it is for the binary object preferences directed case.

Lemma 14. *Clause node C_i holds object α if and only if its variable nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold object β .*

Proof. First, assume that $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold β . By Lemma 9 we know that nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ hold α , and they are C_i 's nearest nodes holding α ; while C_i 's nearest node holding γ is node K . Node's C_i cost for holding α and accessing γ from K is $d_{C_i K} = 1.4$; while the cost for holding γ and accessing α from $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$, is $d_{C_i \bar{L}_{i,j}} = 1.5$. Obviously, node C_i prefers to replicate α .

Now assume that at least one of the nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ holds α . These nodes are C_i 's nearest nodes holding α ; while again C_i 's nearest node holding γ is node K . Node's C_i cost for holding γ and accessing α from $L_{i,j}$, for $j \in \{1, 2, 3\}$, is $d_{C_i L_{i,j}} = 1$; while the cost for holding α and accessing γ from node K is $d_{C_i K} = 1.4$. Obviously, node C_i prefers to replicate γ . \square

Lemma 15. *Node S holds object α if and only if all clause nodes C_1, \dots, C_k hold object γ .*

Proof. First, assume that C_1, \dots, C_k are holding γ . By Lemma 14, S 's nearest node holding α is at least one of $L_{i,j}$ nodes, where $i \in [1, k], j \in \{1, 2, 3\}$; while S 's nearest nodes holding β is node L . The cost for S holding α and accessing β from node L , is $d_{SL} = 2.1$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1, k], j \in \{1, 2, 3\}$, is $d_{SL_{i,j}} = 3$. Obviously, node S prefers to replicate α .

Now assume that at least one of C_1, \dots, C_k holds α . These nodes are S 's nearest node holding α ; while again S 's nearest node holding β is L . The cost for holding β and accessing α from a node C_i , is $d_{SC_i} = 2$; while the cost for holding α and accessing β from a node L is $d_{SL} = 2.1$. Obviously, node S prefers to replicate β . \square

Theorem 16. *There exists an equilibrium for the CSR instance I_3 if and only if node S holds object α .*

Proof. First, assume that S is holding α . By Lemma 15 nodes C_1, \dots, C_k hold object γ , and by Lemma 14 at least one of nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ for each node $C_i, i \in [1, k]$, holds object α , and the corresponding $\bar{L}_{i,j}$ is holding object β . We claim that the placement where A holds γ , B holds γ , and C holds α , is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy.

Node A does not want to deviate since its cost for holding object γ and accessing α from A 's nearest node S , is $d_{AS} = 1$; while the cost for holding object α and accessing γ from A 's nearest node B , is still $d_{AB} = 1$. Node B does not want to deviate since its cost for holding object γ and accessing α from B 's nearest node C , is $d_{BC} = 1$; while the cost for holding object α and accessing γ from B 's nearest node A , is still $d_{BA} = 1$. Node C does not want to deviate since its cost for holding object α and accessing γ from C 's nearest node A , is $d_{CA} = 1$; while the cost for holding object γ and accessing α from C 's nearest node S , is still $d_{CS} = 1$. Also note that none of $S, C_1, \dots, C_k, L_{i,j}, \bar{L}_{i,j}$ for $i \in [1, k], j \in \{1, 2, 3\}$ is getting affected of the objects been held by the gadget nodes.

Now assume that node S holds object β . We are going to prove that for every possible placement over nodes A, B , and C , at least one node wants to deviate from its strategy. Consider the following cases:

- Nodes A, B , and C hold object α : Node B (resp., C) wants to deviate, since the cost for holding object α and accessing γ from B 's (resp., C 's) nearest node C_i , for some $i \in [1, k]$ or from node K , is $d_{BC_i} = 5$ or $d_{BK} = 6.4$ (resp., $d_{CC_i} = 4$ or $d_{CK} = 5.4$); while the cost for holding object γ and accessing α from B 's nearest node A , is $d_{BA} = 2$ (resp., $d_{CA} = 1$).
- Two nodes hold object α and the third holds γ : In the case where A and B hold α , A wants to deviate since the cost while holding α and accessing γ from A 's nearest node C is $d_{AC} = 2$; while the cost for holding γ and accessing α from A 's nearest node B is $d_{AB} = 1$. The other cases are symmetric.
- One node holds α : If A holds α , B wants to deviate since the cost while holding γ and accessing α from B 's nearest node A is $d_{BA} = 2$; while the cost for holding α and accessing γ from B 's nearest node C , is $d_{BC} = 1$. The other cases are symmetric.

- Nodes A , B , and C hold γ : All of them want to deviate. Node A wants to deviate since the cost while holding γ and accessing α from A 's nearest node C_i , for some $i \in [1, k]$, is $d_{AC_i} = 3$; while the cost for holding α and accessing γ from A 's nearest node B is $d_{AB} = 1$. The other cases are symmetric.

Obviously the system does not have a pure Nash equilibrium, which completes the proof. \square

We now show that ϕ is satisfiable if and only if the above CSR games (both undirected and directed cases) (resp., for the binary object preferences, directed case) has a pure Nash equilibrium. Suppose that ϕ is satisfiable and consider a satisfying assignment for ϕ . If the assignment of a variable x_i is True, then we replicate object α in cache of variable node X_i ; otherwise, we replicate object β . By Lemma 9 we know that a variable node X_i holds object α (resp., β) if and only if node \bar{X}_i holds object β (resp., α). In this way we keep the consistency between truth assignment of a variable and its negation. By Lemma 10 (resp., Lemma 14) we know that a clause node C_i , will replicate object β (resp., γ) if and only if at least one of its variable nodes, holds object α . From above, any clause node C_i will hold object β (resp., γ) only if at least one of clause c_i literals is True. By Lemma 11 (resp., Lemma 15), we know that node S , will replicate object α if and only if all clause nodes C_1, \dots, C_k are holding object β (resp., γ). Thus, node S replicates object α only if all clauses c_1, \dots, c_k are True. By Theorems 12 and 13 (resp., 16), we know that there exists a pure Nash Equilibrium if and only if object β is stored to node S ; thus, there exists a pure Nash Equilibrium if and only if all clauses are True. This gives our proof.

6.2 Binary preferences over two objects

Consider the problem 2BIN: does a given CSR instance with two objects and binary preferences possess an equilibrium? We prove that 2BIN is polynomial-time equivalent to the notorious EVEN-CYCLE problem [34]: does a given digraph contain an even cycle? Despite intensive efforts, the complexity of the problem EVEN-CYCLE was open until [23, 29] provided a tour de force polynomial-time algorithm. Our result thus also places 2BIN in P.

Theorem 17. *EVEN-CYCLE is polynomial-time equivalent to 2BIN.*

We prove the polynomial-time equivalence of 2BIN and EVEN-CYCLE by a series of reductions. We first show the equivalence between 2BIN and 2DIR-BIN, which is the sub-class of 2BIN instances in which the node preferences are specified by an unweighted directed graph (henceforth *digraph*); in a 2DIR-BIN instance, we are given a digraph, and the preference of a node for the other nodes increases with decreasing distance in the graph.

Lemma 18. *2BIN is polynomial-time equivalent to 2DIR-BIN.*

Proof. Given a 2BIN instance I with node set V , two objects, node preference relations $\{\geq_i: i \in V\}$, and interest sets $\{S_i: i \in V\}$, we construct a 2DIR-BIN instance I' with the same node set, objects, and interest sets, but with the node preference relations specified by an unweighted digraph G . Our construction will ensure that any equilibrium in I is an equilibrium in I' and vice-versa. For distinct nodes i and j , we have an edge from i to j if and only if j is a most i -preferred node in $V \setminus \{i\}$. We now argue that I has an equilibrium if and only if I' has an equilibrium. A placement for I is an equilibrium if and only if the following holds for each node i : (a) if $|S_i| = 1$, then i holds the lone object in S_i ; (b) if $|S_i| = 2$, then the object not held by i is at an i -most preferred node. Similarly, any equilibrium placement for I' satisfies the following condition for each i : (a) if $|S_i| = 1$, then i holds the lone object in S_i ; (b) if $|S_i| = 2$, then the object not held by i is at a neighbor of i . By our construction of the instances, equilibria of I are equilibria of I' and vice-versa. \square

We next define EXACT-2DIR-BIN, which is the subclass of 2DIR-BIN games where each node is interested in both objects; thus, an EXACT-2DIR-BIN instance is completely specified by a digraph G . We say that a node i is *stable* in a given placement P if P_i is a best response to P_{-i} . We say that an EXACT-2DIR-BIN instance G is *stable* (resp., *1-critical*) if there exists a placement in which all nodes (resp., all nodes except at most one) are stable. Since each node has unit cache capacity, each placement is a 2-coloring of the nodes: think of a node as colored by the object it holds in its cache. Given a placement, an arc is said to be bichromatic if its head and tail have different colors. Note that for any EXACT-2DIR-BIN instance, a node is stable in a placement iff it has a bichromatic outgoing arc.

Lemma 19. *2DIR-BIN and EXACT-2DIR-BIN are polynomial-time equivalent on general digraphs.*

Proof. Since EXACT-2DIR-BIN games are a special subclass of 2DIR-BIN games, we only need to show that 2DIR-BIN games reduce to EXACT-2DIR-BIN games. Given an instance of a 2DIR-BIN game, we need to handle the nodes that are interested in at most one object. First, note that we can remove the outgoing arcs from all such nodes. Let V_0 consist of the nodes with no objects of interest. For each node u in V_0 we add a new node u_0 to V_0 along with arcs (u, u_0) and (u_0, u) . Let red and blue denote the two objects. Let V_r and V_b denote the set of nodes interested in red and blue, respectively. Without loss of generality, let $|V_r| \geq |V_b|$. Add $|V_r| - |V_b|$ additional nodes to the set V_b (so that $|V_r| = |V_b|$) and connect all the nodes in $V_r \cup V_b$ with a directed cycle that alternates strictly between V_r nodes and V_b nodes. The rest of the network is kept the same and all the nodes are set to have interest in both objects. Now, if the original instance is stable then we can stabilize the new instance by having each node in V_r (resp., V_b) cache the red (resp., blue) object, the nodes in V_0 cache any object (so long as an original node u and its associated node u_0 store complementary objects) and the other nodes cache the same object as in the placement that made the original instance stable. And in the other direction, if the transformed instance is stable then in an equilibrium placement, the nodes in V_r must each store an object of one color while each node in V_b stores the object of the other color. By renaming the colors, if necessary, we get a stable coloring (placement) for the original instance. \square

For completeness, we next present some standard graph-theoretic terminology that we will use in our proof. A digraph is said to be *weakly* connected if it is possible to get from a node to any other by following arcs without paying heed to the direction of the arcs. A digraph is said to be *strongly* connected if it is possible to get from a node to any other by a directed path. We will use the following well-known structure result about digraphs: a general digraph that is weakly connected is a directed acyclic graph on the unique set of maximal strongly connected (node-disjoint) components. We will also use the following strengthening of the folklore ear-decomposition of strongly connected digraphs [30]:

Lemma 20. *An ear-decomposition can be obtained starting with any cycle of a strongly connected digraph.*

Proof. The proof is by contradiction. Suppose not, then consider a subgraph with a maximal ear-decomposition obtainable from the cycle in question. If it is not the entire digraph then consider any arc leaving the subgraph. Note that the digraph is strongly connected and hence such an arc must exist. Further, note that every arc in a digraph is contained in a cycle since there is a directed path from the head of the arc to the tail. Starting from the arc follow this cycle until it intersects the subgraph again, as it must because it ends at the tail which lies in the subgraph. This forms an ear that contradicts the maximality of the decomposition. \square

Lemma 21. *EVEN-CYCLE on strongly connected digraphs and EVEN-CYCLE on general digraphs are polynomial-time equivalent.*

Proof. Since strongly connected digraphs are a special subclass of general digraphs it suffices to show that EVEN-CYCLE on general digraphs can be reduced to EVEN-CYCLE on strongly connected digraphs.

Remember that a general digraph has a unique set of maximal strongly connected components that are disjoint and computable in polynomial-time. Further any cycle, including even cycles, must lie entirely within a strongly connected component. Thus a digraph possesses an even cycle iff one of its strongly connected components does. Hence it follows that EVEN-CYCLE on general digraphs reduces to EVEN-CYCLE on strongly connected digraphs. \square

Lemma 22. *EVEN-CYCLE and EXACT-2DIR-BIN games are polynomial-time equivalent on strongly connected digraphs.*

Proof. To show the polynomial-time equivalence, we show that a strongly connected digraph is stable iff it has an even cycle. One direction is easy. If the digraph is stable then consider the placement in which every node is stable. So every node has a bichromatic outgoing arc; by starting at any node and following outgoing bichromatic edges we will eventually loop back on ourselves. The loop so obtained is the required even cycle; it is even because it is composed of bichromatic arcs. In the other direction, if there is an even cycle then we take the ear-decomposition starting with that cycle (Lemma 20), stabilize that cycle (by making each arc bichromatic since it is of even cardinality) and then stabilize each node in each ear by working backwards along the ear. \square

Lemma 23. *Any EXACT-2DIR-BIN game on a strongly connected digraph is 1-critical.*

Proof. Consider an ear-decomposition of the strongly connected digraph starting with a cycle. Observe that all but at most one node of the cycle can be stabilized by arbitrarily assigning one color to a node, and then assigning alternate colors to the nodes as we progress along the cycle. Every node in the cycle, other than possibly the initial node, is stable. The rest of the digraph can be stabilized ear by ear, stabilizing each ear by working backwards from the point of attachment. Hence, all but one node of the digraph can be stabilized. \square

Lemma 24. *EXACT-2DIR-BIN on general digraphs is polynomial-time equivalent to EXACT-2DIR-BIN on strongly connected digraphs.*

Proof. Since strongly connected digraphs are a subclass of general digraphs we need only show that the problem EXACT-2DIR-BIN on general digraphs reduces to EXACT-2DIR-BIN on strongly connected digraphs. A general digraph is stable iff all of its weakly connected components are. A weakly connected component is a directed acyclic graph (dag) on the strongly connected components. It is clear that a weakly connected component cannot be stabilized if any one of the strongly connected components that is a minimal element of the directed acyclic graph cannot be stabilized. Interestingly, the converse is also true. If all of the strongly connected components that are minimal elements of the dag can be stabilized then the entire weakly connected component can be stabilized because each of the other strongly connected components has at least one outgoing arc which is used to stabilize its tail while the rest of the strongly connected component can be stabilized because strongly connected components are 1-critical by Lemma 23. We can determine such a stable placement by processing the strongly connected components in topologically sorted order (according to the dag) starting from the minimal elements. Thus a digraph is stable iff every strongly connected component that is a minimal element is stable. Hence, EXACT-2DIR-BIN on general digraphs is reducible in polynomial-time to strongly connected digraphs. \square

7 Fractional replication games

We introduce a new class of capacitated replication games where nodes can store fractions of objects, as opposed to whole objects, and satisfy an object access request by retrieving enough fractions that make up

the whole object. Rather than associate different identities with different fractions of a given object, we view each portion of an object as being fungible, thus allowing any set of fractions of an object, adding up to at least one, to constitute the whole object. Such fractional replication scenarios naturally arise when objects are encoded and distributed within a network to permit both efficient and reliable access.

Several implementations of fractional replication, in fact, already exist. For instance, fountain codes [4, 31] and the information dispersal algorithm [28] present two ways of encoding an object as a number of smaller pieces – of size, say $1/m$ fraction of the full object size, where m is an integer – such that the full object may be reconstructed from any m of the pieces. A natural formalization is to view each object as a polynomial of high degree, and consider each piece of the object as the evaluation of the polynomial on a random point in a suitable large field. Then, accessing an object is equivalent (with very high probability) to accessing a sufficient number of pieces of the object.

We now present fractional capacitated selfish replication (F-CSR) games, which are an adaptation of the game-theoretic framework developed in Section 4 to fractional replication. We have a set V of nodes sharing a set O of objects. In an F-CSR game, the strategies are *fractional placements*; a fractional placement \tilde{P} is a $|V|$ -tuple $\{\tilde{P}_i : i \in V\}$ where $\tilde{P}_i : O \rightarrow \mathbb{R}$ under the constraint that sum of $\tilde{P}_i(\alpha)$, over all α in O , is at most the cache size of i .

We begin by presenting F-CSR games in the special case of sum utilities, where the generalization from the integral to the fractional setting is most natural. For sum utilities, recall that we are given a cost function d and node-object weights $r_i(\alpha)$, $i \in V$, $\alpha \in O$. Given a fractional global placement \tilde{P} , we define the cost incurred by i for accessing object α as the minimum value of $x_j d_{ij}$ under the constraints that $\sum_j x_j = 1$ and $x_j \leq \tilde{P}_j(\alpha)$ for all j . Then, the total cost incurred by i is the sum, over all objects α , of $r_i(\alpha)$ times the cost incurred by i for accessing α . For a given fractional global placement \tilde{P} , the utility of i is the negative of the total cost incurred by i under \tilde{P} .

We now consider F-CSR games under the more general setting of utility preference relations. As before, each node i has a node preference relation \geq_i and a preference relation \succeq_i among global (integral) placements. Recall that the node and placement preference relations of each node i induce a preorder \sqsubseteq_i among the elements of $O \times (V \setminus \{i\})$ (see Section 4). For F-CSR games, we require the existence of a *total* preorder \sqsubseteq_i , for all i . We now specify the best response function for each player for a given fractional global placement \tilde{P} . For each node i and object α , we determine the assignment $\mu_{i,\tilde{P},\alpha} : V \setminus \{i\} \rightarrow \mathbb{R}$ that is lexicographically minimal under the node preference relation \geq_i subject to the condition that $\mu_{i,\tilde{P},\alpha} \leq \tilde{P}_k(\alpha)$ for each k and $\sum_k \mu_{i,\tilde{P},\alpha}(k) = 1$. We next compute $b_{i,\tilde{P}} : O \times (V \setminus \{i\}) \rightarrow \mathbb{R}$ to be the lexicographically maximal assignment under \sqsubseteq_i subject to the condition that $b_{i,\tilde{P}}(\alpha, k) \leq \mu_{i,\tilde{P},\alpha}(k)$ for all k and $\sum_{\alpha,k} b_{i,\tilde{P}}(\alpha, k)$ is at most the size of i 's cache. The best response of a player i is then to store $\sum_k b_{i,\tilde{P}}(\alpha, k)$ of α in their cache. This completes the definition of F-CSR games.

Using standard fixed-point machinery, we show that every F-CSR game has an equilibrium. We also show that finding equilibria in F-CSR games is PPAD-complete.

Theorem 25. *Every F-CSR instance has a pure Nash equilibrium. Finding an equilibrium in an F-CSR game is PPAD-complete.*

We prove Theorem 25 by establishing separately the existence of equilibria, membership in PPAD, and the PPAD-hardness of finding equilibria.

7.1 Existence of equilibria

Theorem 26. *Every F-CSR instance has a pure Nash equilibrium.*

Proof. By [25], a game has a pure Nash equilibrium if the strategy space of each player is a compact, non-empty, convex space, and the payoff function of each player is continuous on the strategy space of all

players and quasi-concave in the strategy space of the player. In an F-CSR instance, the strategy space of each player i is simply the set of all its fractional placements: that is, the set of functions $f : \mathcal{O} \rightarrow [0, 1]$ subject to condition that $\sum_{\alpha \in \mathcal{O}} f(\alpha) \leq c_i$, where c_i is the cache size of the node (player). The strategy set thus is clearly convex, non-empty, and compact. Furthermore, as defined above, the payoff for any player i under fractional placement \tilde{P} is simply the solution to the following linear program:

$$\begin{aligned} \max - \sum_{\alpha \in \mathcal{O}} r_i(\alpha) \left(\sum_{j \in V} x_{ij}(\alpha) d_{ij} \right) \\ \sum_{j \in V} x_{ij}(\alpha) = 1 \quad \text{for all } i \in V, \alpha \in \mathcal{O} \\ x_{ij}(\alpha) \leq \tilde{P}_j(\alpha) \quad \text{for all } i, j \in V, \alpha \in \mathcal{O} \\ x_{ij}(\alpha) \geq 0 \quad \text{for all } i, j \in V, \alpha \in \mathcal{O} \end{aligned}$$

It is easy to see that the payoff function is both continuous in the placements of all players, and quasi-concave in the strategy space of player i , thus completing the proof of the theorem. \square

7.2 Membership in PPAD

Theorem 27. *Finding an equilibrium in an F-CSR game is in PPAD.*

Proof. Our proof is by a reduction from FSPP (Fractional Stable Paths Problem), which is defined as follows [15]. Let G be a graph with a distinguished destination node d . Each node $v \neq d$ has a list $\pi(v)$ of simple paths from v to d and a preference relation \geq_v among the paths in $\pi(v)$. For a path S , we also define $\pi(v, S)$ to be the set of paths in $\pi(v)$ that have S as a suffix. A *proper suffix* S of P is a suffix of P such that $S \neq P$ and $S \neq \emptyset$. A *feasible fractional paths solution* is a set $w = \{w_v : v \neq d\}$ of assignments $w_v : \pi(v) \rightarrow [0, 1]$ satisfying: (1) **Unity condition**: for each node v , $\sum_{P \in \pi(v)} w_v(P) \leq 1$, and (2) **Tree condition**: for each node v , and each path S with start node u , $\sum_{P \in \pi(v, S)} w_v(P) \leq w_u(S)$.

In other words, a feasible solution is one in which each node chooses at most 1 unit of flow to d such that no suffix is filled by more than the amount of flow placed on that suffix by its starting node. A feasible solution w is *stable* if for any node v and path Q starting at v , one of the following holds: **(S1)** $\sum_{P \in \pi(v)} w_v(P) = 1$, and for each P in $\pi(v)$ with $w_v(P) > 0$, $P \geq_v Q$; or **(S2)** There exists a proper suffix S of Q such that $\sum_{P \in \pi(v, S)} w_v(P) = w_u(S)$, where u is the start node of S , and for each $P \in \pi(v, S)$ with $w_v(P) > 0$, $P \geq_v Q$.

Given an F-CSR G with node set V , object set \mathcal{O} , node preference relations \geq_i for $i \in V$, and utility preference relations \succeq_i for $i \in V$, we construct an instance \mathcal{I} of FSPP as follows. For nodes $i, j \in V$ and object $\alpha \in \mathcal{O}$, we introduce the following FSPP vertices.

- $\text{hold}(i, \alpha)$ representing the amount of α that node i will store in its cache.
- $\text{serve}(i, j, \alpha)$ representing the amount of α that node j will serve for i given a placement for $V \setminus \{i\}$.
- $\text{serve}'(i, j, \alpha)$, an auxiliary vertex needed for $\text{serve}(i, j, \alpha)$.
- $\text{serve}(i, \alpha)$, representing the amount of α that other nodes will serve for i given a placement for $V \setminus \{i\}$.
- $\text{hold}(i)$, representing the best response of i give the placement of other nodes.
- $\text{hold}'(i, \alpha)$, an auxiliary vertex needed for $\text{hold}(i, \alpha)$.

We now present the path sets and preferences for each vertex of the FSPP instance.

- $\text{serve}(i, \alpha)$: the path set includes all paths of the form $\langle \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$, and $\text{serve}(i, \alpha)$ prefers $\langle \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$ over $\langle \text{serve}(i, \alpha), \text{hold}(k, \alpha), d \rangle$ if $j \geq_i k$.
- $\text{serve}'(i, j, \alpha)$: the path set includes all paths of the form $\langle \text{serve}'(i, j, \alpha), \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$ and the direct path $\langle \text{serve}'(i, j, \alpha), d \rangle$. For the preference order, $\text{serve}'(i, j, \alpha)$ prefers all paths $\langle \text{serve}'(i, j, \alpha), \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$ equally, and all of them over the direct path.
- $\text{serve}(i, j, \alpha)$: the path set includes the path $\langle \text{serve}'(i, j, \alpha), d \rangle$ and the direct path $\langle \text{serve}(i, j, \alpha), d \rangle$ with a higher preference for the former path.
- $\text{hold}(i)$: the path set includes paths of the form $\langle \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$, and $\text{hold}(i)$ prefers the path $\langle \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$ over $\langle \text{hold}(i), \text{serve}(i, k, \beta), d \rangle$ if $(j, \alpha) \sqsupseteq_i (k, \beta)$.
- $\text{hold}'(i, \alpha)$: the path set includes paths of the form $\langle \text{hold}'(i, \alpha), \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$ all of which are preferred equally, and the direct path $\langle \text{hold}'(i, \alpha), d \rangle$ which is preferred the least.
- $\text{hold}(i, \alpha)$: the path set includes two paths $\langle \text{hold}'(i, \alpha), d \rangle$ and the direct path with a higher preference for the former path.

We now show that the F-CSR instance has an equilibrium if and only if the FSPP instance has an equilibrium. Our proof is by giving a mapping f from global fractional placements in the F-CSR instance to feasible solutions in the FSPP instance such that (a) if \tilde{P} is an equilibrium for the F-CSR instance, then $f(\tilde{P})$ is an equilibrium for the FSPP instance, and (b) if w is an equilibrium for the FSPP instance, then $f^{-1}(w)$ is an equilibrium for the F-CSR instance.

Let \tilde{P} denote any fractional placement of the F-CSR instance. We now define the solution $f(\tilde{P})$ of the FSPP instance. In $f(\tilde{P})$ vertex $\text{hold}(i, \alpha)$ plays $\tilde{P}_i(\alpha)$ on the direct path and $1 - \tilde{P}_i(\alpha)$ on the other path in its path set, for every i in V and α in O . The remaining vertices play their best responses, considered in the following order. First, consider vertices of the form $\text{serve}(i, \alpha)$. In the best response, the amount played by $\text{serve}(i, \alpha)$ on the path $\langle \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$, equals $\mu_{i, \tilde{P}, \alpha}(j)$; recall that $\mu_{i, \tilde{P}, \alpha}(j)$ is the assignment that is lexicographically minimal under the node preference relation \geq_i subject to the condition that $\mu_{i, \tilde{P}, \alpha} \leq \tilde{P}_k(\alpha)$ for each k and $\sum_k \mu_{i, \tilde{P}, \alpha}(k) = 1$. We next consider the vertices of the form $\text{serve}'(i, j, \alpha)$. In its best response, vertex $\text{serve}'(i, j, \alpha)$ plays $\mu_{i, \tilde{P}, \alpha}(j)$ on the path $\langle \text{serve}'(i, j, \alpha), \text{serve}(i, \alpha), \text{hold}(j, \alpha), d \rangle$. Next, in its best response, vertex $\text{serve}(i, j, \alpha)$ plays $\mu_{i, \tilde{P}, \alpha}(j)$ on its direct path and $1 - \mu_{i, \tilde{P}, \alpha}(j)$ on its remaining path. We now consider the best response of vertex $\text{hold}(i)$; it distributes its unit among paths of the form $\langle \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$ (for all j in $V \setminus \{i\}$ and α in O) lexicographically maximally under the total preorder \sqsupseteq_i over node-object pairs. That is, $\text{hold}(i)$ plays $b_{i, \tilde{P}}(\alpha, j)$ on the path $\langle \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$. We next consider the best response of the vertex $\text{hold}'(i, \alpha)$; it plays $1 - \sum_j b_{i, \tilde{P}}(\alpha, j)$ on its direct path and $b_{i, \tilde{P}}(\alpha, j)$ on the path $\langle \text{hold}'(i, \alpha), \text{hold}(i), \text{serve}(i, j, \alpha), d \rangle$. This completes the definition of the solution $f(\tilde{P})$.

We now argue that if \tilde{P} is an equilibrium so is $f(\tilde{P})$. By construction, every vertex other than of the form $\text{hold}(i, \alpha)$ play their best responses in $f(\tilde{P})$. We next show that i plays a best response in \tilde{P} if and only if the vertices $\text{hold}(i, \alpha)$ play their best response in $f(\tilde{P})$. The best response of $\text{hold}(i, \alpha)$ is to play $1 - \sum_j b_{i, \tilde{P}}(\alpha, j)$ on the path $\langle \text{hold}(i, \alpha), \text{hold}'(i, \alpha), d \rangle$ and the $\sum_j b_{i, \tilde{P}}(\alpha, j)$ on its direct path. The best response of i in \tilde{P} is to set $\tilde{P}_i(\alpha)$ to $\sum_j b_{i, \tilde{P}}(\alpha, j)$. Thus if \tilde{P} is an equilibrium, then so is $f(\tilde{P})$. Furthermore, if w is an equilibrium, by definition of f , $\tilde{P} = f^{-1}(w)$ is well-defined. Since the best responses of i and the vertices $\text{hold}(i, \alpha)$ are consistent, \tilde{P} is also an equilibrium. This completes the reduction from F-CSR to FSPP, placing F-CSR in PPAD. \square

7.3 PPAD-Hardness

This section is devoted to the proof of the following theorem.

Theorem 28. *The problem of finding an equilibrium in F-CSR games is PPAD-hard even when the underlying cost function d is a metric.*

Our reduction is from preference games [15]. Given a preference game G with n players $1, 2, \dots, n$ and their preferences given by \geq_i , we construct an F-CSR game \hat{G} as follows. The game \hat{G} has a set V of $n^2 + 3n$ players numbered 1 through $n^2 + 3n$, and a set O of $2n$ objects $\alpha_1, \dots, \alpha_{2n}$. We set the utility function for each node to be the sum utility function, thus ensuring that the desired monotonicity and consistency conditions are satisfied.

We next present the metric cost function d over the nodes. We group the players into four sets $V_1 = \{i : 1 \leq i \leq n\}$, $V_2 = \{i \cdot n + j : 1 \leq i \leq n, 1 \leq j \leq n\}$, $V_3 = \{n^2 + n + i : 1 \leq i \leq n\}$, and $V_4 = \{n^2 + 2n + i : 1 \leq i \leq n\}$. For each node i in V_1 and j in V_3 , we set $d_{ii} = 2$ and $d_{ij} = 4$. We set $d_{n^2+n+i, n^2+2n+i} = 3$. For each node i in V_1 and $k = i \cdot n + j$, we set d_{ik} as follows: if $j >_i i$ then d_{ij} equals $6 - \ell/n$ when j is the ℓ th most preferred player for i ; if $i \geq_i j$, then d_{ij} equals 1. All the other distances are obtained by using metric properties.

We finally specify the object weights. For $k \in V_1$, we set $r_k(\alpha_i) = 1$ for all $i \neq k$ such that $i \geq_k k$; we set $r_k(\alpha_k) = 2.5$ such that $4 < 2r_k(\alpha_k) \leq 5$. For node $k = i \cdot n + j$ in V_2 , we set $r_k(\alpha_j) = 1$. For node $k = n^2 + n + i$ in V_3 , we set $r_k(\alpha_i) = r_k(\alpha_{i+n}) = 1$. Finally, for node $k = n^2 + 2n + i$ in V_4 , we set $r_k(\alpha_{i+n}) = 1$.

Given a placement P for \hat{G} , we define a solution $\omega(P) = \{w_{ij}\}$ for the preference game G : $w_{ij} = P_i(\alpha_j)$. The following lemma immediately follows from the definition of \hat{G} .

Lemma 29. *The following statements hold for any placement P for \hat{G} .*

- For $k = i \cdot n + j$, $1 \leq j \leq n$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_j) = 1$.
- For $k = n^2 + n + i$, $1 \leq i \leq n$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_{n+i}) = 1$.
- For $k = n^2 + n + i$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_i) = 1 - P_i(\alpha_i)$ and $P_k(\alpha_{n+i}) = P_i(\alpha_i)$.

Lemma 30. *Let P be a placement for \hat{G} in which every node not in V_1 plays their best response. Then, the best response of a node i in V_1 is the lexicographically maximum $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \dots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \dots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$. \square*

Proof. Fix a node i in V_1 . By Lemma 29, node $i \cdot n + j$ holds object j , for $1 \leq j \leq n$; each of these nodes is at distance at least 5 and at most 6 away from i . By Lemma 29, for every node $k = n^2 + n + j$, $1 \leq j \leq n$, $P_k(\alpha_j) = 1 - P_j(\alpha_j)$ and $P_k(\alpha_{n+j}) = P_j(\alpha_j)$.

We now consider the best response of node i . We first note that for any $j \in \{1, \dots, n\} \setminus \{i\}$ such that $i \geq_i j$, $P_i(\alpha_j) = 0$ since the nearest full copy of α_j is nearer than the nearest node holding any fraction of object α_i . Let S denote the set of j such that $j \geq_i i$. For any j in $S \setminus \{i\}$, $P_i(\alpha_j) \leq P_j(\alpha_j)$ since node $n^2 + n + j$ at distance 5 holds $1 - P_j(\alpha_j)$ fraction of α_j , the nearest node holding any fraction of α_i is at distance 4, and $4r_i(\alpha_i) > 5r_i(\alpha_j)$. Furthermore, for any j, k in S if $j >_i k$, then the farthest $P_j(\alpha_j)$ fraction of α_j is farther than the farthest $P_k(\alpha_k)$ fraction of α_k , implying that in the best response, if $P_i(\alpha_j) < P_j(\alpha_j)$ then $P_i(\alpha_k) = 0$. Thus, the best response of i is the unique lexicographically maximum solution $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \dots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \dots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$. \square

Lemma 31. *A placement P is an equilibrium for \widehat{G} if and only if $\omega(P)$ is a equilibrium for G and every node not in V_1 plays their best response in P .*

Proof. Consider an equilibrium placement P for \widehat{G} . Clearly, every node plays their best response. We now prove that $\omega(P)$ is an equilibrium for G . Fix a node i in V_1 . By Lemma 30, the best response of i is the unique lexicographically maximum solution $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \dots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \dots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$. Since this applies to every node i , it is immediate from the definitions of $\omega(P)$ and preference games that if P is an equilibrium for \widehat{G} then $\omega(P)$ is an equilibrium for G .

We now consider the reverse direction. Suppose we have a placement P in which every player not in V_1 plays their best response and $\omega(P)$ is an equilibrium for the preference game G . By Lemma 30 and the definition of $\omega(P)$, the best response of i in G matches that in the F-CSR game; hence every player in V_1 also plays their best response in P , implying that P is an equilibrium for \widehat{G} . \square

The construction of \widehat{G} from G is clearly polynomial time. Furthermore, given any equilibrium for \widehat{G} , an equilibrium for G can be constructed in linear time. We thus have a reduction from a PPAD-complete problem to F-CSR implying that the latter is PPAD-hard, thus completing the proof of Theorem 28.

8 Concluding remarks

In this paper we have defined integral and fractional selfish replication games (CSR and F-CSR) in networks, where the cache capacity of each node is bounded and all objects are of uniform size. We have shown that a pure Nash equilibrium can be computed for every hierarchical network, using a new notion of fictional players. In general, we have almost completely characterized the complexity of CSR games: For what classes of games do equilibria exist? Can we determine efficiently whether they exist? When they do exist, can we efficiently find them? One complexity question that is still open is the case of undirected networks with binary preferences. We conjecture that finding equilibria in such games (which we prove are potential games) is PLS-hard. In general, we would like to study the convergence of the best response process for the cases of games where equilibria exist.

We showed that F-CSR games always have equilibria, though they may be hard to find. It is not hard to argue that an equilibrium in the corresponding integral variant is an equilibrium in the fractional instance. So whenever an “integral” equilibrium can be determined efficiently, so can a “fractional” equilibrium. An interesting direction of research is to identify other special cases of fractional games where equilibria may be efficiently determined. We also note that our proof of existence of equilibria in F-CSR games, currently presented for the case of unit-size objects, extends to arbitrary object sizes.

Finally, we note that our model assumes that the sets of nodes, objects, and preference relations are all static. We believe our results will be meaningful for environments where these sets change infrequently. Developing better models for addressing more dynamic scenarios is an important practical research direction.

References

- [1] E. Angel, E. Bampis, G. G. Pollatos, and V. Zissimopoulos. Optimal data placement on networks with constant number of clients. *CoRR*, abs/1004.4420, 2010.
- [2] K. Arrow. *Social Choice and Individual Values*. Yale University Press, 1951.
- [3] I. D. Baev, R. Rajaraman, and C. Swamy. Approximation algorithms for data placement problems. *SIAM J. Comput.*, 38(4):1411–1429, 2008.

- [4] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. In *SIGCOMM '98*, pages 56–67, 1998.
- [5] X. Chen, X. Deng, and S.-H. Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM)*, 56(3), 2009.
- [6] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C. H. Papadimitriou, and J. Kubiawicz. Selfish caching in distributed systems: a game-theoretic analysis. In *PODC*, pages 21–30, 2004.
- [7] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *STOC ACM*, pages 71–78, 2006.
- [8] N. R. Devanur, N. Garg, R. Khandekar, V. Pandit, A. Saberi, and V. V. Vazirani. Price of anarchy, locality gap, and a network service provider game. In *WINE*, pages 1046–1055, 2005.
- [9] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 347–351, New York, NY, USA, 2003. ACM Press.
- [10] D. Fudenberg and D. Levine. *The Theory of Learning in Games*. MIT Press, 1998.
- [11] M. Garey and D. Johnson. *Computers and intractability*. Freeman Press, 1979.
- [12] M. X. Goemans, L. Li, V. S. Mirrokni, and M. Thottan. Market sharing games applied to content distribution in ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(5):1020–1033, 2006.
- [13] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988.
- [14] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. P. Abstract. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *STOC ACM*, pages 654–663, 1997.
- [15] S. Kintali, L. J. Poplawski, R. Rajaraman, R. Sundaram, and S.-H. Teng. Reducibility among fractional stability problems. *FOCS*, 2009.
- [16] M. Korupolu, C. G. Plaxton, and R. Rajaraman. Placement algorithms for hierarchical cooperative caching. *Journal of Algorithms*, 38:260–302, 2001.
- [17] M. R. Korupolu and M. Dahlin. Coordinated placement and replacement for large-scale distributed caches. *IEEE Trans. Knowl. Data Eng.*, 14(6):1317–1329, 2002.
- [18] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *STACS*, pages 404–413, March 1999.
- [19] N. Laoutaris, G. Smaragdakis, A. Bestavros, and I. Stavrakakis. Mistreatment in distributed caching groups: Causes and implications. In *INFOCOM*, 2006.
- [20] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, and A. Bestavros. Distributed placement of service facilities in large-scale networks. In *INFOCOM*, pages 2144–2152, 2007.
- [21] N. Laoutaris, O. Telelis, V. Zissimopoulos, and I. Stavrakakis. Distributed selfish replication. *IEEE Trans. Parallel Distrib. Syst.*, 17(12):1401–1413, 2006.

- [22] A. Leff, J. L. Wolf, and P. S. Yu. Replication algorithms in a remote caching architecture. *IEEE Trans. Parallel Distrib. Syst.*, 4(11):1185–1204, Nov 1993.
- [23] W. McCuaig, N. Robertson, P. D. Seymour, and R. Thomas. Permanents, pfaffian orientations, and even directed circuits. In *STOC*, pages 402–405, 1997.
- [24] N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.
- [25] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.
- [26] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *JCSS*, 48(3):498–532, 1994.
- [27] G. G. Pollatos, O. Telelis, and V. Zissimopoulos. On the social cost of distributed selfish content replication. In *Networking*, pages 195–206, 2008.
- [28] M. O. Rabin. Efficient dispersal of information for security, load balancing and fault tolerance. *Journal of the ACM*, 36:335–348, 1989.
- [29] N. Robertson, P. D. Seymour, and R. Thomas. Permanents, pfaffian orientations, and even directed circuits. *Annals of Mathematics*, pages 929–975, 1999.
- [30] A. Schrijver. *Combinatorial Optimization (3 Vol.)*. Springer-Verlag Berlin Heidelberg, 2003.
- [31] A. Shokrollahi. Raptor codes. In *IEEE Trans Inf Theory*, pages 2551–2567, 2006.
- [32] R. Tewari, M. Dahlin, H. M. Vin, and J. S. Kay. Design considerations for distributed caching on the internet. In *ICDCS*, pages 273–284, 1999.
- [33] O. Wolfson, S. Jajodia, and Y. Huang. An adaptive data replication algorithm. *ACM Transactions on Database Systems*, 22:255–314, 1997.
- [34] D. H. Younger. Graphs with interlinked directed circuits. In *Proceedings of Midwestern Symposium on Circuit Theory*, volume 2, pages XVI2.1–XVI2.7, 1973.